S V E U Č I L I Š T E U Z A G R E B U

ELEKTROTEHNIČKI FAKULTET - ZAGREB

MARIO ŽAGAR

OSNOVE DIGITALNIH RAČUNALA

ZBIRKA RIJEŠENIH ZADATAKA

ZAGREB, 1979.

21298

S V E U Č I L I Š T E U Z A G R E B U

ELEKTROTEHNIČKI FAKULTET - ZAGREB

MARIO ŽAGAR

OSNOVE DIGITALNIH RAČUNALA

ZBIRKA RIJEŠENIH ZADATAKA

ZAGREB 1979.

Odobrenje Odbora za izdavačku djelatnost Skupštine Sveučilišta u Zagrebu

Cijena za studente Sveučilišta u Zagrebu
Din 44.–

Tisak: Sveučilišna naklada Liber–Zagreb, Trg Maršala Tita 14
Broj: 989/SL

Ova zbirka zadataka namijenjena je studentima Elektrotehničkog fakulteta za lakše savladavanje gradiva iz grupe predmeta "Organizacija računarskih sustava".

Za rješavanje zadataka i njihovo razumijevanje, potrebno je proučiti knjigu prof. dr G. Smiljanića: "Osnove digitalnih računala".

Zbirka je podijeljena na 4 grupe zadataka:

- brojni sustavi, kodovi i zapisivanje podataka u računalo
- elementarno školsko računalo
- povezivanje školskog računala s ulazno/izlaznim jedinicama
- rješavanje problema kod računala sa zadanim karakteristikama.

Zahvaljujem se svima koji su mi pomogli u pripremanju zbirke zadataka i onima koji će svojim primjedbama i sugestijama omogućiti da slijedeća izdanja budu još bolja.

U Zagrebu, siječnja 1979. Autor

S A D R Ž A J :

strana

1. Brojni sustavi, kodovi i zapisivanje podataka u računalo 1

2. Elementarno školsko računalo 17

3. Povezivanje školskog računala s ulazno/izlaznim jedinicama (U/I programiranje).. 49

4. Rješavanje problema kod računala sa zadanim karakteristikama 79

4.1 Hipotetsko računalo 79

4.2 Komercijalno miniračunalo (PDP-8) 86

4.3 Mikroračunala:

Fairchild F8 90
Intel I8080 94
Intel I8085 95
Zilog Z80 99

1. BROJNI SUSTAVI, KODOVI I ZAPISIVANJE PODATAKA U RAČUNALO

1. Pretvori binarni broj u oktalni:

   a) 011010101111_2

   b) 11011001_2

   c) 1111101110010001_2

Rješenje:

a) $011010101111_2 = \underbrace{011}_{3}\ \underbrace{010}_{2}\ \underbrace{101}_{5}\ \underbrace{111}_{7}\ _2 = 3257_8$

(binarni broj grupira se po tri znamenke s desna na lijevo)

b) $11011001_2 = \underbrace{11}_{3}\ \underbrace{011}_{3}\ \underbrace{001}_{1}\ _2$

c) $1111101110010001_2 = \underbrace{1}_{1}\ \underbrace{111}_{7}\ \underbrace{101}_{5}\ \underbrace{110}_{6}\ \underbrace{010}_{2}\ \underbrace{001}_{1}\ _2 =$

$= 175621_8$

2. Pretvori oktalni broj u binarni:

   a) 7776_8

   b) 31_8

   c) 4365_8

   d) 176543_8

Rješenje:

a) $7776_8 = \underbrace{111}_{7}\ \underbrace{111}_{7}\ \underbrace{111}_{7}\ \underbrace{110}_{6}\ _2 = 111111111110_2$

b) $31_8 = \underbrace{011}_{3}\ \underbrace{001}_{1}{}_2 = 011001_2 = 11001_2$ (8)

c) $4365_8 = \underbrace{100}_{4}\ \underbrace{011}_{3}\ \underbrace{110}_{6}\ \underbrace{101}_{5}{}_2 = 100011110101_2$ (8)

d) $176543_8 = \underbrace{001}_{1}\ \underbrace{111}_{7}\ \underbrace{110}_{6}\ \underbrace{101}_{5}\ \underbrace{100}_{4}\ \underbrace{011}_{3}{}_2 = 1111110101100011_2$ (8)

3. Pretvori binarni broj u decimalni:

a) 1011_2

b) 111_2

c) 11011001_2

Rješenje:

a) $1011_2 = 1\cdot 2^0 + 1\cdot 2^1 + 0\cdot 2^2 + 1\cdot 2^3 =$

$= 1\cdot 1 + 1\cdot 2 + 0\cdot 4 + 1\cdot 8 =$

$= 1 + 2 + 0 + 8 = 11_{10}$

b) $111_2 = \overset{(2^2)}{1}\ \overset{(2^1)}{1}\ \overset{(2^0)}{1} = 1\cdot 4 + 1\cdot 2 + 1\cdot 1 = 7_{10}$

c) $11011001_2 = \overset{128}{1}\ \overset{64}{1}\ \overset{32}{0}\ \overset{16}{1}\ \overset{8}{1}\ \overset{4}{0}\ \overset{2}{0}\ \overset{1}{1}{}_2 =$

$= 128 + 64 + 0 + 16 + 8 + 0 + 0 + 1 = 217_{10}$

($2^0 = 1_{10}$ $\quad$ $2^8 = 256_{10} = 1/4$ K

$2^1 = 2_{10}$ $\quad$ $2^9 = 512_{10} = 1/2$ K

$2^2 = 4_{10}$ $\quad$ $2^{10} = 1024_{10} = 1$ K

$2^3 = 8_{10}$ $\quad$ $2^{11} = 2048_{10} = 2$ K

$2^4 = 16_{10}$ $\quad$ $2^{12} = 4096_{10} = 4$ K

$2^5 = 32_{10}$ $\quad$ $2^{13} = 8192_{10} = 8$ K

$2^6 = 64_{10}$ $\quad$ $2^{14} = 16384_{10} = 16$ K

$2^7 = 128_{10}$ $\quad$ $2^{15} = 32768_{10} = 32$ K

$2^{16} = 65536_{10} = 64$ K)

4. Pretvori decimalni broj u binarni:

a) 11_{10}

b) 217_{10}

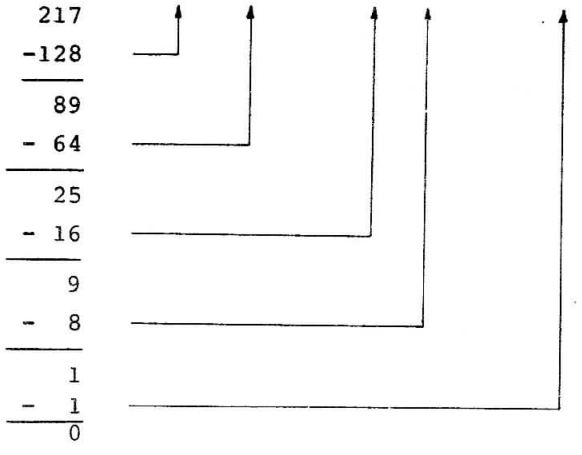
c) 127_{10}

d) 512_{10}

Rješenje:

a) 11_{10} = (16 8 4 2 1) 0 1 0 1 1 = 1011_2

b) 217_{10} = (128 64 32 16 8 4 2 1) 1 1 0 1 1 0 0 1 = 11011001_2



c) $127_{10} = \overbrace{\begin{array}{cccccccc} 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}}^{127_{10}} = 1111111_2$

d) $512_{10} = \begin{array}{cccccccccc} 512 & 256 & 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} =$

$= 1000000000_2$

5. Pretvori binarni broj u heksadecimalni:

a) 011010101111_2

b) 11011001_2

c) 100010100_2

Rješenje:

a) $011010101111_2 = \underbrace{0110}_{6}\ \underbrace{1010}_{A}\ \underbrace{1111}_{F}\ 2 = 6AF_{16}$

b) $11011001_2 = \underbrace{1101}_{D}\ \underbrace{1001}_{9}\ 2 = D9_{16}$

c) $100010100_2 = \underbrace{1}_{1}\ \underbrace{0001}_{1}\ \underbrace{0100}_{4}{}_2 = 114_{16}$

6. Pretvori heksadecimalni broj u binarni:

a) 32_{16}

b) $A2_{16}$

c) $DDFE_{16}$

Rješenje:

a) $32_{16} = \underbrace{3}_{0011}\ \underbrace{2}_{0010}\ {}_{16} = {}_{2} = 00110010_2 = 110010_2$

b) $A2_{16} = \underbrace{1010}_{A}\ \underbrace{0010}_{2}\ {}_{2} = {}_{16} = 10100010_2$

c) $DDFE_{16} = \underbrace{1101}_{D}\ \underbrace{1101}_{D}\ \underbrace{1111}_{F}\ \underbrace{1110}_{E}\ {}_{2} = {}_{16} = 1101110111111110_2$

(bin.	dec.	heksadec.	okt.)
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	8	8	10
1001	9	9	11
1010	10	A	12
1011	11	B	13
1100	12	C	14
1101	13	D	15
1110	14	E	16
1111	15	F	17

7. Pretvori heksadecimalni broj u decimalni:

a) 32_{16}

b) $A2_{16}$

c) $15A_{16}$

Rješenje:

a) $32_{16} = \underbrace{0011}_{3}\ \underbrace{0010}_{2}\ {}_{2} = {}_{16} = 32 + 16 + 2 = 50_{10}$

ili

$32_{16} = 3 \cdot 16^1 + 2 \cdot 16^0 = 48_{10} + 2_{10} = 50_{10}$

b) $A2_{16} = \begin{matrix} 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{matrix} =$

$= 128 + 32 + 2 = 162_{10}$

ili

$A2_{16} = 10 \cdot 16^1 + 2 \cdot 16^0 = 160 + 2 = 162_{10}$

c) $15A_{16} = \begin{matrix} 256 & 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{matrix} =$

$= 256 + 64 + 16 + 8 + 2 = 346_{10}$

ili

$15A_{16} = 1 \cdot 16^2 + 5 \cdot 16^1 + 10 \cdot 16^0 = 256+80+10 = 346_{10}$

8. Pretvori heksadecimalni broj u oktalni:

a) 32_{16}

b) $A2_{16}$

c) $15A_{16}$

Rješenje:

a) $32_{16} = \underbrace{0011}_{3}\ \underbrace{0010}_{2}{}_{16} = \underbrace{00}_{0}\ \underbrace{110}_{6}\ \underbrace{010}_{2}{}_{8} = 62_8$

b) $A2_{16} = \underbrace{1010}_{A}\ \underbrace{0010}_{2}{}_{16} = \underbrace{10}_{2}\ \underbrace{100}_{4}\ \underbrace{010}_{2}{}_{8} = 242_8$

c) $15A_{16} = \underbrace{0001}_{1}\ \underbrace{0101}_{5}\ \underbrace{1010}_{A}{}_{16} = \underbrace{000}_{0}\ \underbrace{101}_{5}\ \underbrace{011}_{3}\ \underbrace{010}_{2}{}_{8} = 532_8$

9. Pretvori decimalni broj u heksadecimalni:

a) 11_{10}

b) 32_{10}

c) 250_{10}

d) 1024_{10}

Rješenje:

a) $11_{10} = B_{16}$

ili

$$11_{10} = \begin{matrix} 8 & 4 & 2 & 1 \\ 1 & 0 & 1 & 1 \end{matrix}_2 = B_{16}$$

b) $32_{10} = \begin{matrix} 256 & 16 & 1 \\ 0 & 2 & 0 \end{matrix}_{16} = 20_{16}$

ili

$$32_{10} = \underbrace{\begin{matrix} 32 & 16 \\ 1 & 0 \end{matrix}}_{2} \underbrace{\begin{matrix} 8 & 4 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{matrix}}_{0}{}_2 = 20_{16}$$

16

c) $250_{10} = \begin{matrix} 256 & 16 & 1 \\ 0 & 15 & 10 \end{matrix} = FA_{16}$

ili

$$250_{10} = \begin{matrix} 256 & 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 0 & \underbrace{1 & 1 & 1 & 1}_{F} & \underbrace{1 & 0 & 1 & 0}_{A} \end{matrix}_2 = FA_{16}$$

16

d) $1024_{10} = \begin{matrix} 256 & 16 & 1 \\ 4 & 0 & 0 \end{matrix}_{16} = 400_{16}$

10. Pretvori oktalni broj u heksadecimalni:

a) 777_8

b) 242_8

c) 1024_8

d) 10000_8

Rješenje:

a) $$777_8 = \underbrace{111}_{7}\ \underbrace{111}_{7}\ \underbrace{111}_{7_8}\,{}_2 = \underbrace{1}_{1}\ \underbrace{1111}_{F}\ \underbrace{1111}_{F_{16}}\,{}_2 = 1FF_{16}$$

b) $$242_8 = \underbrace{010}_{2}\ \underbrace{100}_{4}\ \underbrace{010}_{2_8}\,{}_2 = 0\ \underbrace{1010}_{A}\ \underbrace{0010}_{2_{16}}\,{}_2 = A2_{16}$$

kompliciranije je:

$$242_8 = 2\cdot 8^2 + 4\cdot 8^1 + 2\cdot 8^0 = 128 + 32 + 2 = 162_{10} =$$

$$\begin{matrix} & 256 & 16 & 1 & \\ = & 0 & A & 2 & _{16} \end{matrix} = A2_{16}$$

c) $$1024_8 = 001\ 000\ 010\ 100\ {}_2 = 0010\ 0001\ 0100_2 = 214_{16}$$

d) $$10000_8 = 001\ 000\ 000\ 000\ 000_2 =$$

$$= 0001\ 0000\ 0000\ 0000_2 = 1000_{16}$$

11. Pretvori zadani decimalni broj u binarni:

$0{,}34375_{10}$

Rješenje:

$0{,}34375_{10}$	$\times\ 2 = 0{,}6875_{10}$	cjelobrojni dio je 0
$0{,}6875_{10}$	$\times\ 2 = 1{,}375_{10}$	" " " 1
$0{,}375_{10}$	$\times\ 2 = 0{,}75_{10}$	" " " 0
$0{,}75_{10}$	$\times\ 2 = 1{,}5_{10}$	" " " 1
$0{,}5_{10}$	$\times\ 2 = 1{,}0_{10}$	" " " 1

dakle:

$$0{,}34375_{10} = \overset{2^0}{0},\ \overset{2^{-1}}{0}\ \overset{2^{-2}}{1}\ \overset{2^{-3}}{0}\ \overset{2^{-4}}{1}\ \overset{2^{-5}}{1}_2 = 0{,}01011_2$$

12. Pretvori binarni broj u decimalni:

a) $0{,}01011_2$

b) $1{,}111_2$

Rješenje:

a) $0{,}01011_2 = 0\cdot 2^0 + 0\cdot 2^{-1} + 1\cdot 2^{-2} + 0\cdot 2^{-3} +$

$+ 1\cdot 2^{-4} + 1\cdot 2^{-5} = 0 + 0 + 0{,}25 + 0 +$

$+ 0{,}0625 + 0{,}03125 = 0{,}34375_{10}$

b) $1{,}111_2 = 1\cdot 2^0 + 1\cdot 2^{-1} + 1\cdot 2^{-2} + 1\cdot 2^{-3} =$

$= 1 + 0{,}5 + 0{,}25 + 0{,}125 = 1{,}875_{10}$

13. Pretvori decimalni broj u heksadecimalni :

$0{,}631_{10}$

Rješenje:

$0{,}631_{10} \times 16 = 10{,}\underbrace{096}_{10}$ cjelobrojni dio je $10_{10} = A_{16}$

$0{,}\underbrace{096}_{10} \times 16 = 1{,}\underbrace{536}_{10}$ " " " $1_{10} = 1_{16}$

$0{,}\underbrace{536}_{10} \times 16 = 8{,}\underbrace{576}_{10}$ " " " $8_{10} = 8_{10}$

$0{,}\underbrace{576}_{10} \times 16 = 9{,}216_{10}$ " " " $9_{10} = 9_{16}$

.

.

itd.

konverzija na 4 znamenke daje:

$0{,}631_{10} = 0{,}A189_{16}$

Ako želimo veću točnost , moramo nastaviti s konverzijom !

ZADACI ZA VJEŽBU (bez rješenja):

14. Pretvori slijedeće brojeve u decimalne:

a) 110101011_2
b) $0,10111101_2$
c) $11010,1011010_2$
d) 134726_8
e) $0,165124_8$
f) $1776,03521_8$
g) $16A93E_{16}$
h) $0,ABC04_{16}$
i) $A13E,D19082_{16}$

15. Pretvori slijedeće brojeve u binarne:

a) 37941_{10}
b) $0,193721_{10}$
c) $369,17362_{10}$
d) 134726_8
e) $0,165124_8$
f) $1776,03521_8$
g) $16A93E_{16}$
h) $0,ABC04_{16}$
i) $A13E,D19082_{16}$

16. Pretvori slijedeće brojeve u oktalne:

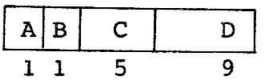
a) 110101011_2
b) $0,10111101_2$
c) $11010,1011010_2$
d) 37941_{10}
e) $0,193721_{10}$
f) $369,17362_{10}$
g) $16A93E_{16}$
h) $0,ABC04_{16}$
i) $A13E,D19082_{16}$

17. Pretvori slijedeće brojeve u heksadecimalne:

a) 110101011_2
b) $0,10111101_2$
c) $11010,1011010_2$
d) 134726_8
e) $0,165124_8$
f) $1776,03521_8$
g) 37941_{10}
h) $0,193721_{10}$
i) $369,17362_{10}$

18. Provjeri dobivene rezultate iz zadataka 14.,15.,16. i 17. (npr. tako da odgovore iz zadatka 15.a) do f) pretvoriš u heksadec. i usporediš s rezultatima zadatka 17. d) do i)).

19. Napiši decimalni broj u binarnom obliku (u prikazu s pokretnim zarezom), ako je riječ računala organizirana na slijedeći način:



bitova

A = predznak mantise
B = predznak eksponenta
C = eksponent
D = mantisa

Baza sistema je 10. Da li je navedena organizacija riječi zadovoljavajuća? Brojevi koje treba napisati su : a) 123_{10}
b) $-2,3_{10}$

Rješenje:

Navedena organizacija riječi nije zadovoljavajuća jer je premali broj bitova za prikaz mantise. To se može riješiti na taj način da se više riječi u računalu koristi za prikaz jednog broja(vidi: G.Smiljanić: Osnove digitalnih računala, 1.10.5).

a) $+123_{10} = (+)\ 0,123\ \ 10^{(+)3}$

A D B C

A = 0 (+)

B = 0 (+)

C = $3_{10} = 00011_2$

D = $123_{10} = 1111011_2 = 001111011_2$

Sadržaj riječi u računalu je dakle:

A B C D

0 0 00011 001111011$_2$ = 0000011001111011$_2$

b) $-2,3_{10} = -\ 0,23 \cdot 10^{+1}$

A = 1_2 (-)

B = 0_2 (+)

C = $1_{10} = 00001_2$

D = $23_{10} = 10111_2 = 000010111_2$

rezultat je dakle: 1000001000010111$_2$

20. Napravi "jedan" i "dva" komplement od slijedećih brojeva:

a) 10110_2

b) 11111_2

c) 00000_2

Rješenje:

"jedan komplement" dobijemo tako da zamijenimo 0 i 1.

a) "jedan kompl." od 10110_2 je 01001_2

b) " " " 11111_2 " 00000_2

c) " " " 00000_2 " 11111_2

"dva komplement" dobijemo tako da "jedan" komplementu broja dodamo 1_2 .

a) "dva kompl." od 10110_2 je 01001_2

$$\begin{array}{r} + \quad 1_2 \\ \hline 01010_2 \end{array}$$

b) " " " 11111_2 je 00001_2

c) " " " 00000_2 " 00000_2

21. Zbroji dva binarna broja :

a) $100110_2 + 1110_2$

b) $1011_2 + 100_2$

c) $1111_2 + 11_2$

Rješenje:

a)
$$\begin{array}{rr} {\scriptstyle 111} & {\scriptstyle 1} \\ 100110_2 & = 38_{10} \\ + \quad 1110_2 & =+ \ 14_{10} \\ \hline 110100_2 & 52_{10} \end{array}$$

b)
$$\begin{array}{rrr} 1011_2 & = 11_{10} & = B_{16} \\ + \ 100_2 & = + \ 4_{10} & = + \ 4_{16} \\ \hline 1111_2 & = 15_{10} & = F_{16} \end{array}$$

c)
$$\begin{array}{rcrcr} 1111_2 & = & 15_{10} & = & F_{16} \\ +\ 11_2 & = & +\ 3_{10} & = & +\ 3_{16} \\ \hline 10010_2 & = & 18_{10} & = & 12_{16} \end{array}$$

22. Oduzmi dva binarna broja:

a) $100110_2 - 1110_2$

b) $1110_2 - 100110_2$

Rješenje:

a) $100110_2 - 1110_2 = 100110_2 + (-1110_2) =$

(negativan broj dobijemo iz zadanog broja pomoću "dva komplementa")

$= 100110_2 + \underbrace{110010_2}_{\text{"dva kompl" od } 1110_2} = \boxed{1}\,011000_2 = 011000_2$

ako je preliv jednak "1", znači da je odbijanje ispravno izvršeno !

dakle:

$$\begin{array}{rcrcr} 100110_2 & \Rightarrow & 100110_2 & = & 38_{10} \\ -\ 1110_2 & & +\ 110010_2 & = & -14_{10} \\ \hline & & \boxed{1}\,011000_2 & = & 24_{10} \end{array}$$

b) $1110_2 - 100110_2 = 1110_2 + (-100110_2) =$

$= 1110_2 + \underbrace{(011010_2)}_{\text{"dva kompl." od } 100110_2} = \boxed{0}\,101000_2$

nema preliva! Dobivenu vrijednost treba komplementirati ("dva kompl.") da dobijemo ispravan rezultat (predznak je negativan).

$$\begin{array}{r} 1110_2 \\ -\ 100110_2 \\ \hline \end{array} \Longrightarrow \begin{array}{r} 1110_2 \\ +\ 011010_2 \\ \hline \underline{0}\,101000_2 \end{array} \qquad \begin{array}{l} = 14_{10} \\ = -38_{10} \\ \hline = -24_{10} \end{array}$$

" dva kompl." od 101000_2 je 011000_2 dakle:

$$-\ 011000_2 = -\ 24_{10}$$

23. Pomnoži dva binarna broja:

a) $100110_2 \times 1110_2$

b) $1001_2 \times 11_2$

c) $1001_2 \times 10_2$

Rješenje:

a) $\overbrace{100110_2}^{38_{10}} \times \overbrace{1110_2}^{14_{10}}$

$$\begin{array}{r} 000000 \\ 100110 \\ 100110 \\ 100110 \\ \hline 1000010100_2 \end{array} = 532_{10}$$

b) sukcesivnim zbrajanjem

$$\begin{array}{rl} 1001_2 \times 11_2 = & 1001_2 \\ & 1001_2 \\ & +\ 1001_2 \\ \hline & 11011_2 \end{array}$$

c) $1001_2 \times 10_2$

10010_2 množenje s bazom znači pomak u lijevo !

24. Podijeli dva binarna broja:
$100110_2 : 1110_2$

Rješenje:

$100110_2 : 1110_2 = 10_2$

$-\ 1110_2$

001010_2

ostatak

$38_{10} : 14_{10} = 2_{10}$

-28

10_{10}

ostatak

ZADACI BEZ RJEŠENJA:

25. Zbroji slijedeće brojeve:

a) $1010011_2 + 111101_2$
b) $101,11010_2 + 11,00110_2$
c) $0,1011101_2 + 0,1100_2$
d) $0,1011101_2 + 11,00110_2$

Izvrši provjeru tako da zadane binarne brojeve pretvoriš u decimalne i izvršiš zbrajanje !

26. Pretvori slijedeće brojeve u binarne i izvrši računske operacije:

a) $38_{10} - 19_{10}$
b) $27_{10} - 52_{10}$
c) $21,5_{10} - 31,32_{10}$
d) $26_8 - 12_8$
e) $A372_{16} - 1B39_{16}$
f) $6,4_8 - 12,33_8$

27. Napiši "jedan" i "dva" komplement slijedećih brojeva:

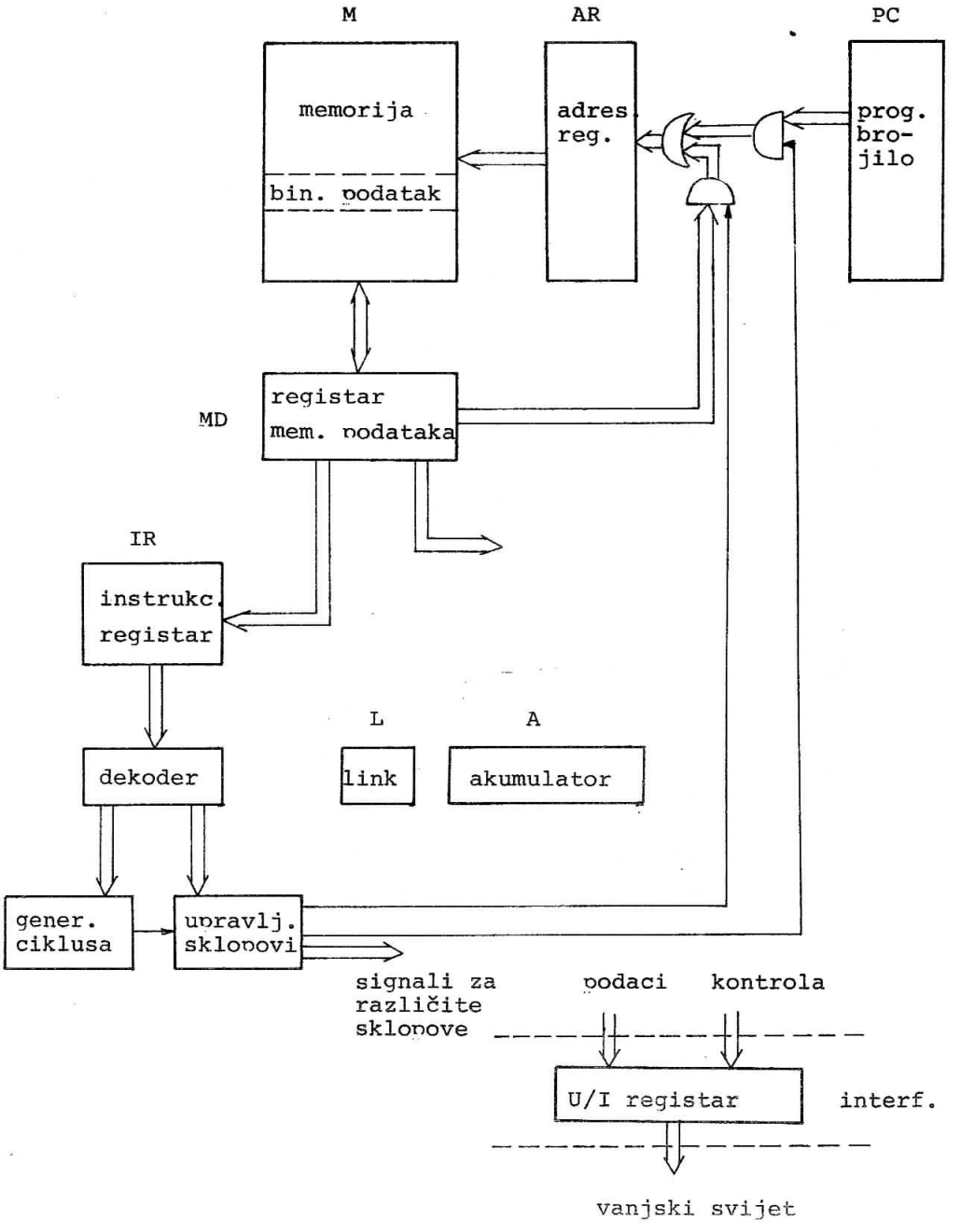
a) 1011_2
b) 23_8
c) $2A_{16}$

2. ŠKOLSKO RAČUNALO

(Karakteristike "školskog računala" i instrukcioni set opisani su u knjizi G. Smiljanić: "Osnove digitalnih računala")

28. Skicirati i objasniti osnovne dijelove računala (hardverske elemente) !

Rješenje:



29. Navesti osnovne registre u računalu i opisati njihovu funkciju!

Rješenje:

- akumulator (A), primarni dio aritmetičko logičke jedinice za smještaj binarne informacije. Aritm. i log. operacije izvode se tako da je jedan operand (najčešće) u akumulatoru, a drugi je negdje izvan akumulatora (konstanta, reg., memorija) i rezultat operacije je (najčešće) spremljen u akumulatoru.

- programsko brojilo (PC) je registar koji pokazuje memorijsku adresu instrukcije koja se treba izvršiti. Izvršavanjem tekuće instrukcije sadržaj programskog brojila se uveća za 1.

- link (L) predstavlja jednobitni nastavak akumulatora, a omogućava i povezivanje najnižeg i najvišeg bita u "cirkulirajuću" memoriju (lijevo, desno).

- adresni registar (AR) pokazuje adresu memorijske lokacije s kojom se vrši neka akcija.

- memorijski registar podataka (MD) služi za prihvat podataka iz memorijske lokacije koju adresira AR registar (ili za upis podataka u tu memorijsku lokaciju)

- instrukcioni registar (IR) otkriva o kojoj je instrukciji riječ.

- generator stanja (GS) na temelju sadržaja instrukcionog registra generira niz impulsa za obavljanje odredjene operacije (svi ti impulsi sinhronizirani su sa CLOCK-om).

30. Izvršiti podjelu instrukcija u školskom računalu:
    a) prema onome na što se odnose,
    b) prema onome što čine !

Rješenje:

a) podjela instrukcija prema onome na što se odnose:
    1) instrukcije koje se odnose na memoriju (dva operanda)
    2) instrukcije koje se odnose na registre (najčešće akumulator)
    3) ulazno/izlazne instrukcije

b) podjela instrukcija prema onome što čine:
    1) aritmetičko-logičke instrukcije i instrukcije za prijenos podataka
    2) kontrolne instrukcije
    3) instrukcije za povezivanje s potprogramima
    4) operacije s jednim operandom
    5) komunikacija ulaz/izlaz

31. Opiši tok izvodjenja instrukcije "STORE M". Koji je sadržaj akumulatora i memorijske lokacije M prije i poslije izvodjenja instrukcije (oktalno) ?

	(PC)	
a)	200	STORE 50
b)	30	STORE 101
c)	152	STORE 152

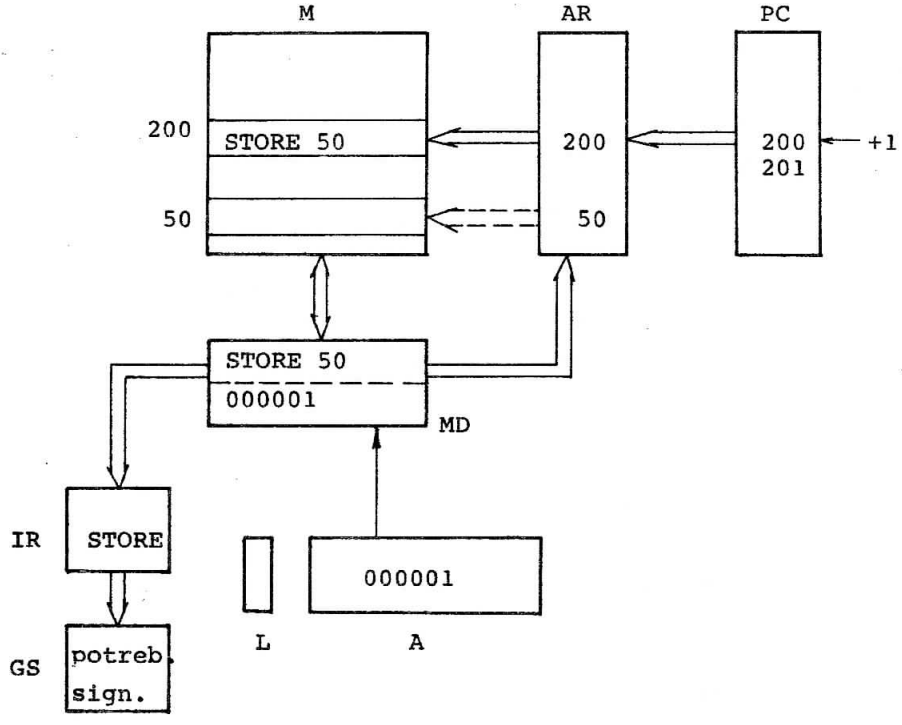
Rješenje:

(A) ⟶ M

oblik instrukcije

0 011 ... ...

kod adresa M



a)

	(PC)	(A)	(mem. lok. 50)
prije	000200_8	000001_8	002221_8
poslije	000201_8	000001_8	000001_8

b)

			(mem. lok. 101)
prije	000030_8	002271_8	000001_8
poslije	000031_8	002271_8	002271_8

c)

	(PC)	(A)	(mem. lok. 152)
prije	000152_8	000012_8	030152_8 - STORE 152
poslije	000153_8	000012_8	000012_8

(M) i (A) prije izvodjenja instrukcije je proizvoljan !

32. Opiši tok izvodjenja instrukcije " ADD M " . Koji je sadržaj akumulatora i memorijske lokacije prije i poslije izvodjenja instrukcije (oktalno) :

a) 200 ADD 50
b) 30 ADD 101
c) 152 ADD 152 ?

Rješenje:

(A) + (M) → A

oblik instrukcije

0 001 ... ...

kod | adresa M

a)

	(PC)	(A)	(mem. lok. 50)
prije	000200_8	006210_8	000100_8
poslije	000201_8	006310_8	000100_8

b)

			(mem. lok. 101)
prije	000030_8	002271_8	000001_8
poslije	000031_8	002272_8	000001_8

c)

			(mem. lok. 152)
prije	000152_8	000012_8	010152_8 - ADD 152
poslije	000153_8	010164_8	010152_8

(A) i (M) prije izvodjenja instrukcije je proizvoljan !

33. Opiši tok izvodjenja instrukcije "AND M". Koji je sadržaj akumulatora i memorijske lokacije prije i poslije izvodjenja instrukcije (oktalno):

a) 200 AND 50
b) 30 AND 101
c) 152 AND 152 ?

Rješenje:

(A) · (M) → A

oblik instrukcije:

0 000 ... ...
kod adresa M

a)

	(PC)	(A)	(mem. lok. 50)
prije	000200_8	006210_8	002221_8
poslije	000201_8	002200_8	002221_8

			(mem. lok. 101)
b) prije	000030_8	002271_8	000001_8
poslije	000031_8	000001_8	000001_8

			(mem. lok. 152)
prije	000152_8	000012_8	000152_8 - AND 152
poslije	000153_8	000012_8	000152_8

(A) i (M) prije izvodjenja instrukcije je proizvoljan !

34. Opiši tok izvodjenja instrukcije " JUMP M ". Koji je sadržaj akumulatora i programskog brojila (PC) prije i poslije izvodjenja instrukcije (oktalno) :

a) 200 JUMP 50
b) 30 JUMP 101
c) 152 JUMP 152 ?

Rješenje:

$M \longrightarrow PC$

oblik instrukcije:

0 101 ... ...

kod adresa skoka

		(PC)	(A)
a)	prije	000200_8	000002_8
	poslije	000050_8	000002_8
b)	prije	000030_8	177777_8
	poslije	000101_8	177777_8
c)	prije	000152_8	101010_8
	poslije	000152_8	101010_8

(A) prije izvodjenja instrukcije je proizvoljan !

35. Opiši tok izvodjenja instrukcije "ISZ M". Koji je sadržaj akumulatora, memorijske lokacije i programskog brojila prije i poslije izvodjenja instrukcije (oktalno):

a) 200 ISZ 50
b) 30 ISZ 101
c) 152 ISZ 152 ?

Rješenje:

$(M) + 1 \longrightarrow M$

$M = 0 \Rightarrow I+2$

$M \neq 0 \Rightarrow I+1$

oblik instrukcije:

0 010 ... ...

kod adresa M

		(PC)	(A)	(mem. lok. 50)
a)	prije	000200_8	000000_8	000001_8
	poslije	000201_8	000000_8	000002_8

b)		(PC)	(A)	(mem. lok. 101)
	prije	000030_8	010001_8	177777_8
	poslije	000032_8	010001_8	000000_8

c)				(mem. lok. 152)
	prije	000152_8	177777_8	020152_8 - ISZ 152
	poslije	000153_8	177777_8	020153_8 - ISZ 153

(A) i (M) u a) i b)dijelu zadatka izabrani su proizvoljno !

36. Opiši tok izvodjenja instrukcije "JMS M". Koji je sadržaj akumulatora, memorijske lokacije i programskog brojila prije i poslije izvodjenja instrukcija (oktalno):
    a) 200 JMS 50
    b) 30 JMS 101
    c) 152 JMS 152

Rješenje:

(PC) → M

I ⇒ M + 1

oblik instrukcije:

0 100 ... ...

kod adresa

a)		(PC)	(A)	(mem. lok. 50)
	prije	000200_8	111072_8	000400_8
	poslije	000051_8	111072_8	000201_8

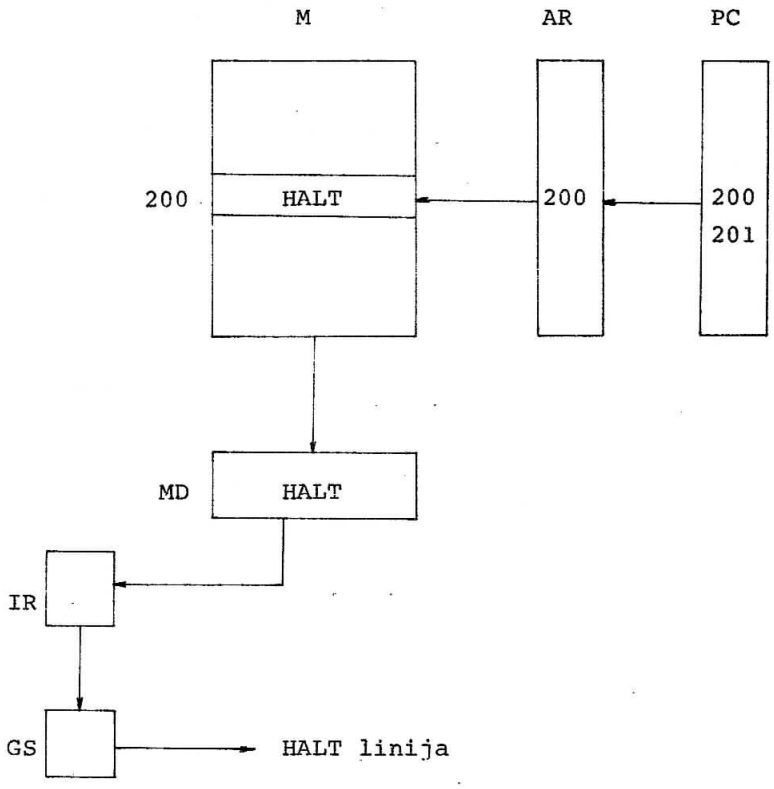
b)				(mem. lok. 101)
	prije	000030_8	111072_8	000000_8
	poslije	000102_8	111072_8	000031_8

c)				(mem. lok. 152)
	prije	000152_8	111072_8	000001_8
	poslije	000153_8	111072_8	000153_8

(A) prije izvodjenja instrukcije je proizvoljan !

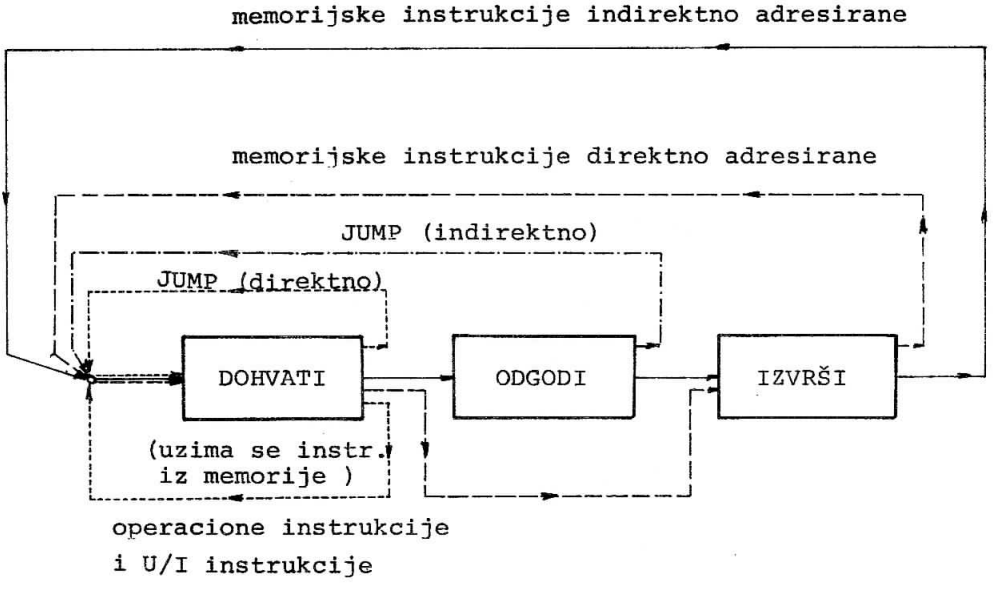
37. Skicirati tok izvodjenja operacionih instrukcija!

Rješenje: (npr. instrukcija HALT)



38. Navesti bitne karakteristike (razlike) izmedju memorijskih i operacionih instrukcija !

39. Podijeliti instrukcije prema broju ciklusa potrebnih za njihovo izvršavanje ! Koje se najvažnije operacije odvijaju u pojedinom ciklusu ?



40. Analizirati sadržaje glavnih registara nakon izvršavanja pojedinih ciklusa instrukcije. Zadana je instrukcija, lokacija memorije u kojoj se ta instrukcija nalazi i početno stanje pojedinih registara. (oznaka "x" znači da sadržaj tog registra nije bitan).

Rješenje: (sve numeričke vrijednosti su oktalne)

mem. lok.	instrukcija	poč. stanje	DOBAVI	ODGODI	IZVRŠI
650 651	JMS I.+1 000100	(PC)=650 (MA)= x (MD)= x	651 650,651 JMS I.+1	651 100 100	101 100 651
500 50	STORE 50 001000	(PC)=500 (MA)= x (MD)= x (A)=000004	501 500, 50 STORE 50 000004	/	501 50 4 4
500 50	STORE I 50 001000	(PC)= 500 (MA)= x (MD)= x (A)=000003	501 500, ,50 STORE I 50 000003	501 1000 1000 3	501 1000 3 3
300 301	ISZ 301 177777	(PC)=300 (MA)= x (MD)= x (A)= 100	301 300,301 ISZ 301 100	/	302 301 0 100
50	CLEAR	(PC)= 50 (MA)= x (MD)= x (A) = 50	51 50, 50 CLEAR 000000	/	/
20	JMS 10	(PC)= 20 (MA)= ~~X~~ (MD)= X	21 20 JMS10 Rješenje		

41. Potrebno je promijeniti predznak sadržaju mem. lokacije 10_8 . Napiši program i njegov mašinski kod (binarno i heksadecimalno). Program neka počinje od lokacije 2_8 !

Rješenje:

adresa mem. lokacije (okt.)	simbolički program	mašinski kod bin.	heksa.
0	.	.	.
1	.	.	.
2	CLEAR	0111010000000000	7400
3	ADD 10	0001000000001000	1008
4	COMPL	0111000100000000	7100
5	IAC	0111000000001000	7008
6	STORE 10	0011000000001000	3008
7	HALT	0111100000010000	7810
10	X	X	X
.	.	.	.

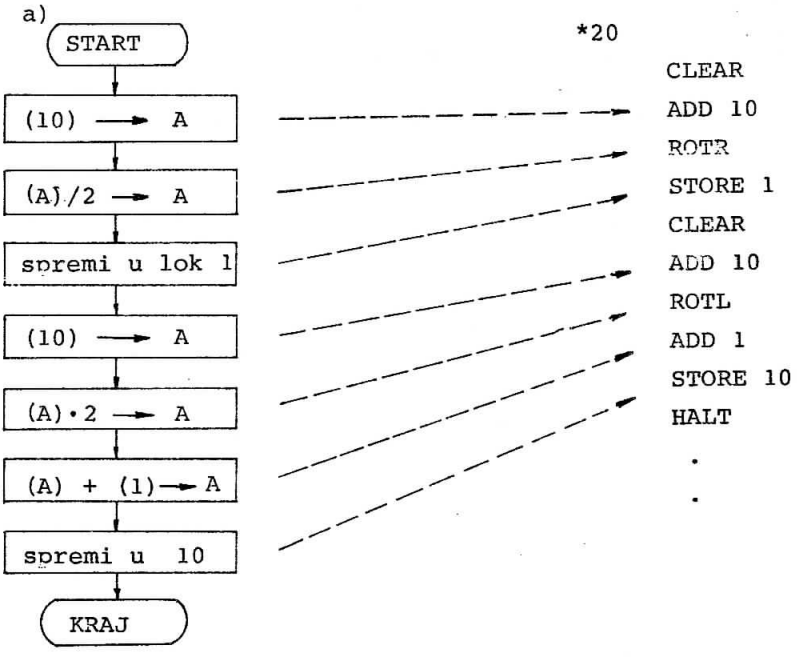
(U lokaciji 10 je sadržaj "X" kojem treba promijeniti predznak).

42. a) Nacrtaj dijagram toka i napiši program za množenje broja s konstantom 2,5. Broj koji se množi uzima se iz mem. lokacije 10_8 . Rezultat se sprema takodjer u lokaciju 10_8. Za privremeno spremanje medjurezultata može se koristiti lokacija 1. Program počinje od lokacije 20_8 .

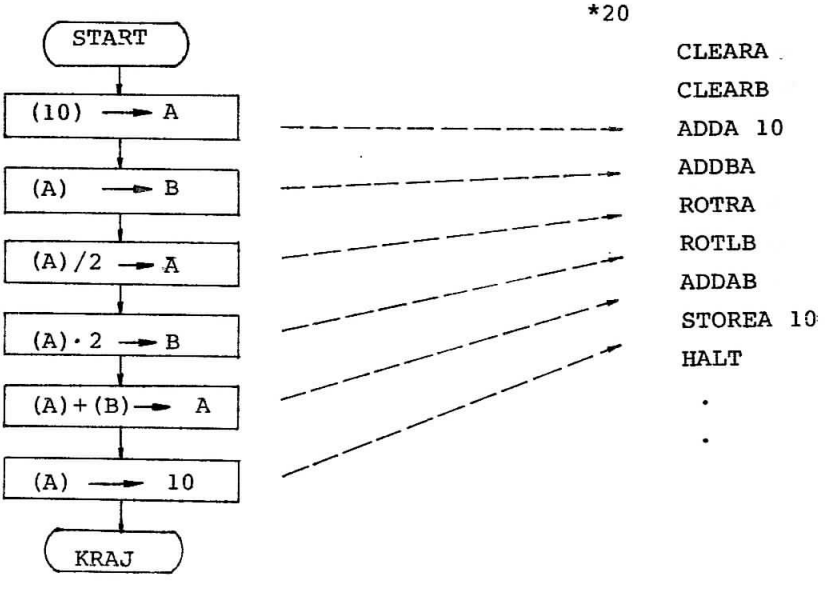
b) Pretpostavimo da je sve isto kao u prethodnom primjeru,a razlika je jedino u tome da računalo ima 2 akumulatora. Na raspolaganju su sve instrukcije školskog računala s time da sve instrukcije koje se odnose na akumulator imaju dodatnu oznaku o kojem se akumulatoru radi (npr. CLEARA - očisti akumulator A, ADDB 10 - dodaj akumulatoru B sadržaj memorijske lokacije 10_8).

Rješenje:

a)


START
(10) → A
(A)/2 → A
spremi u lok 1
(10) → A
(A)•2 → A
(A) + (1) → A
spremi u 10
KRAJ
*20
CLEAR
ADD 10
ROTR
STORE 1
CLEAR
ADD 10
ROTL
ADD 1
STORE 10
HALT

b)


START
(10) → A
(A) → B
(A)/2 → A
(A)•2 → B
(A)+(B) → A
(A) → 10
KRAJ
*20
CLEARA
CLEARB
ADDA 10
ADDBA
ROTRA
ROTLB
ADDAB
STOREA 10
HALT

(RAZLIČITI NAČINI ADRESIRANJA)

43. Napiši program koji radi slijedeću operaciju:

$\frac{(150)+(152)}{2} \longrightarrow 152$. Program počinje od lokacije 100_8 . Koristi se direktnim apsolutnim adresiranjem !

Rješenje:

```
*100
        CLEAR
        ADD  150
        ADD  151        direktno
        ROTR            apsolutno
        STORE 152       adresiranje
          .
          .
          .

150     000004
151     000002
152     000003
```

44. Napiši program koji radi slijedeću operaciju:

$\frac{(150)+(151)}{2} \longrightarrow 152.$ Program počinje od lokacije 100_8 . Koristi se direktnim relativnim adresiranjem s obzirom na trenutni sadržaj memorijskog adresnog registra (MA) !

Rješenje:

```
                .
                .
                .
100     CLEAR
101     ADD .+47  ------> (MA)+47(8) ------> MA
102     ADD .+47
103     ROTR
104     STORE .+46
                .
                .
                .

150     000004
151     000002
153     000003
                .
                .
```

(svi brojevi pisani su oktalno).

45. Napiši program koji radi slijedeću operaciju:

$\frac{(150)+(151)}{2}$ ⟶ 152. Program neka počinje od lokacije 100_8 . Koristi se indirektnim adresiranjem preko lokacija 106_8 , 107_8 i 110_8. Napisati mašinski kod za zadani program. (Svi brojevi pišu se oktalno)!

Rješenje:

mašinski kod	adresa	simb. instr.	
072000	100	CLEAR	
110106	101	ADD I 106	indir.
110107	102	ADD I 107	apso-
070100	103	ROTR	lutno adr.

(nastavak na idućoj strani)

(nastavak)

mašinski kod	adresa	simb. instr.	
130110	104	STORE I .+4	→indirektno
074020	105	HALT	(relativno)
000150	106	000150	adresiranje
000151	107	000151	
000152	110	000152	
.	.	.	
.	.	.	
.	.	.	
000004	150	000004	
000002	151	000002	
000003	152	000003	

46. U koje će memorijske lokacije biti upisana "0" nakon što se izvede slijedeći program ? (programske petlje, adresna modifikacija).

```
*300
      CLEAR
301   STORE 150
      ISZ 301
      JUMP 301
      HALT
        .
        .
```

Rješenje:

"0" će biti upisana u memorijske lokacije 150_8 do 301_8 . Iza toga program mijenja operacioni kod nekih svojih instrukcija pa daljnje akcije računala nisu pod kontrolom ovog programa !

47. Potrebno je premjestiti 18_{10} riječi iz memorijskog bloka s početnom adresom 350_8 u blok s početnom adresom 200_8 koristeći tehniku adresne modifikacije. Program počinje od lokacije 320_8 . Kodiraj oktalno zadani program !

Rješenje:

*320		okt. kod	
320	CLEAR	072000	
	ADD 350	010350	
	STORE 200	030200	
	ISZ 321	020321	
	ISZ 322	020322	
	ISZ 330	020330	
	JUMP 320	050320	
	HALT	074020	
330	177756	177756	-18_{10}

(SIMBOLIČKO ADRESIRANJE)

48. Napisati i analizirati program koji simulira logičku I funkciju :

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

Rješenje:

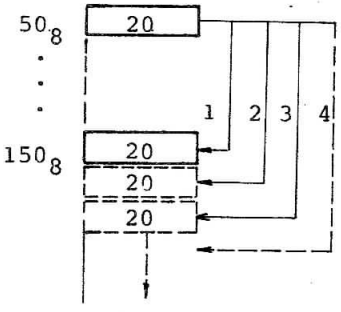
```
POC   CLEAR          JED   CLEAR
      ADD X                ADD Y
      SKIPZ                STORE Z
      JUMP JED             HALT
      STORE Z        X     000000
      HALT           Y     000001
                     Z     000000
```

49. Premjestiti 20_{10} brojeva iz memorijskog bloka s početnom adresom 50_8 u blok s početnom adresom 150_8. Program napisati tako da se može izvoditi više puta bez dodatnih intervencija. Koristiti simboličko adresiranje. Koje lokacije treba promijeniti ako želimo druge početne adrese blokova i drugi broj podataka ?

Rješenje:

```
                .
                .
                .
POCET           CLEAR
                ADD A
                STORE LOK
                CLEAR
                ADD B
                COMPL
                IAC
                STORE COUNT
DO              CLEAR
                ADD I LOK
                STORE I NLOK
                ISZ LOK
                ISZ NLOK
                ISZ COUNT
                JUMP DO
                HALT
A               000050
B               000024
COUNT           000000   , 177754₈ -24₈
LOK             000000   , 000050₈
NLOK            000150
```

Ako se program izvodi više puta od lokacije "POCET", rezultat je:



adresa i broj podataka koji se prenose

upisuje se programski

adresa kamo se prenose podaci

(POTPROGRAMI)

50. Napiši otvoreni potprogram (makroinstrukciju) za izračunavanje apsolutne vrijednosti broja. Broj se nalazi u akumulatoru.

Rješenje:

```
          *100
          ABS   MACRO
ime potprograma SKIPN
          JUMP  .+3
          COMPL
          IAC
          END MACRO
```

ovo asembler prepiše na mjestu gdje je MACRO ABS pozvan !

```
          .
          .
          .
```

glavni program

```
          POC   CLEAR
                ADD BROJ
                ABS
                STORE ABROJ
                .
                .
                .
          BROJ  000020
          ABROJ 000000
                END
```

➢ oznaka asembleru da je kraj prevodjenja u mašinski kod !

51. U programu se često pojavljuje kombinacija instrukcija

```
CLEAR
ADD M
```

kojom se prebacuje sadržaj mem. lokacije "M" u akumulator. Napisati makroinstrukciju "LOAD" koja će zamijeniti prethodnu kombinaciju u simboličkom pisanju programa !

Rješenje:

```
LOAD  MACRO  M1
      CLEAR
      ADD M1
      END MACRO
```

Koristeći rješenje iz prethodnog zadatka možemo napisati glavni program:

```
*100
POCET   LOAD BROJ
        ABS
        STORE ABROJ
          .
          .
          .
          .
BROJ    101010
ABROJ   000000
        END
```

U memoriji je zapisan mašinski kod za slijedeće instrukcije:

```
CLEAR
ADD BROJ
SKIPN
JUMP .+3
COMPL
IAC
STORE ABROJ
  .
  .
```

52. Treba napisati zatvoreni potprogram za izračunavanje apsolutne vrijednosti broja. Broj se unosi iz programa u potprogram preko akumulatora !

Rješenje:

```
ABS     0
        SKIPN
        JUMP I ABS
        COMPL
        IAC
        JUMP I ABS
          .
          .
```

glavni program:

```
          .
          .
        CLEAR
        ADD BROJ
        JMS ABS
        STORE ABROJ
          .
          .
        HALT
BROJ    000001
ABROJ   000000
          .
```

53. Odrediti sadržaj lokacija 100_8, 200_8, 400_8 i 401_8 nakon izvodjenja slijedećeg programa. Koliko je ciklusa potrebno da se program završi (napisati uz svaku instrukciju koliko ciklusa traje)? (Svi brojevi su oktalni!)

		broj ciklusa:
77	CLEAR	1
100	ADD 100	2
101	ADD 100	2
102	STORE I 105	3
103	JUMP I 105	2
104	0	
105	000401	
.		
.		
200	0	
.		
.		
400	0	
401	0	2
402	HALT	1

Rješenje: (100) = 010100
(200) = 000001
(400) = 000000
(401) = 020200 ISZ 200

54. Analiziraj program i odredi sadržaj lokacije X nakon izvodjenja programa, ako je početni sadržaj lokacije X jednak 3 .

```
*10
      CLEAR
      IAC
      COMPL
      ADD X
      STORE X
X     000003
```

Rješenje:
(X) = 000001

55. Odredi koja će slijedeća instrukcija biti izvršena ako program počinje od lokacije 100_8 !

```
        .                          .
        .                          .
100     JMS I 101          477     CLEAR
101     000500             500     IAC
        .                  501     COMPL
        .                          .
                                   .
```

Rješenje: instrukcija u lokaciji 501_8 .

56. Potrebno je testirati vrijednost najznačajnijeg bita akumulatora. Ako je (A)=0 nastavi program, ako je (A)=1 zaustavi program !

Rješenje:

```
        .
        .
        .
CLEARL
ROTL
SKIPL
SKIP
HALT
        .
        .
```

jednostavnije je:

```
        .
        .
        .
SKIPP
HALT
        .
        .
```

57. Odredi koju će lokaciju dostići slijedeći program:

```
        .
        .
100     ISZ 101
101     JMS I 102
102     000500
103     000600
104     000700
```

Rješenje: 601_8

58. Odredi u koju će zadnju lokaciju skočiti program !

a)
```
600  JMS 650
      .
      .
650  JUMP 602
651  JUMP I 650
      .
      .
```

Rješenje: 601_8

b)
```
150  JMS I 160
      .
      .
160  000161
161  JUMP I 160
162  JUMP I 161
      .
      .
```

Rješenje: 151_8

59. U kojoj će se lokaciji zaustaviti zadani program ?
Program počinje od lokacije 610_8 !

```
      .                        .
      .                        .
610  JMS 160             157  HALT
611  ISZ 160             160  JMS I 161
612  JMS I 160           161  JUMP I 160
613  JUMP 157            162  HALT
614  JUMP 610                  .
615  HALT                      .
      .                  Rješenje: 157₈
      .
```

60. Upotpuni potprogram IF koji će ispitati sadržaj akumulatora i usmjeriti daljnji tok programa na jednu od tri adrese napisane ispod naredbe skoka u potprogram!

```
        .
        .
    JMS IF
    JUMP A1   / skoči u A1 ako je (A) manji od 0
    JUMP A2   /   "     A2  "   "  (A) veći od  0
    JUMP A3   /   "     A3  "   "  (A) jednak   0
        .
        .
        .
IF      0
        STORE AKU
        CLEAR
        ADD I IF
        STORE JP
        CLEAR
        ADD AKU
        SKIPN
        SKIP
        JUMP I JP
        SKIPZ
        SKIP
        ISZ JP
        ISZ JP
        JUMP I JP
AKU     0
JP      0
```

61. Potprogram IF1 analizira sadržaj akumulatora i upućuje program u lokacije A, A+1 ili A+2. Upotpuni program i označi u kojim će se lokacijama nastaviti program za svaki od mogućih slučajeva ?

```
    .                          .
    .                          .
JMS IF1                   IF1   0
JUMP A                          SKIPZ
    .                           SKIP
    .                           ISZ I IF1
                                SKIPN
                                ISZ I IF1
                                JUMP I IF1
                                  .
                                  .
```

Rješenje:

za	(A) > 0	A + 1
za	(A) = 0	A + 2
za	(A) < 0	A

62. Treba napisati program koji će odrediti broj negativnih i broj pozitivnih sadržaja lokacija 1000_8 do 1077_8!

Rješenje:

```
        *500
POC     CLEAR
        ADD CONST
        STORE COUNT
        CLEAR
        ADD OPERND          priprema
        STORE AX
        CLEAR
        STORE N
        STORE P

NOVI    CLEAR
        ADD I AX
        SKIPN
        JUMP POZ
```

(nastavak na idućoj strani)

(nastavak)

```
        ISZ N
INC     ISZ AX
        ISZ COUNT
        JUMP NOVI
        HALT
POZ     ISZ P
        JUMP INC
CONST   177700
COUNT     0
OPERND  001000
AX        0
P         0
N         0
        END
```

63. Nadopuniti program kojim se simulira rad JK bistabila !

J	K	Q^{n+1}
0	0	Q^n
0	1	0
1	0	1
1	1	$\bar{Q}^n$

Rješenje:

```
        *1000
POC1    CLEAR
        ADD J
        SKIPZ
        JUMP BIST
        ADD K
        SKIPZ
        <------ SKIP
        HALT
        CLEAR
        STORE Q
        HALT
BIST    CLEAR
        ADD K
        <------ SKIPZ
```

(nastavak na idućoj strani)

(nastavak)

```
        JUMP .+4
        IAC
        STORE Q
        ←——— HALT
        CLEAR
        ADD Q
        COMPL
        AND MASK
        STORE Q
        HALT
J       000000
K       000001
Q       000000
MASK    000001
```

64. Napiši program za zbrajanje dva broja dvostruke točnosti !

Rješenje:

```
        *0
ZBR     CLEAR
        CLEARL
        ADD X1
        ADD Y1
        STORE Z1
        CLEAR
        ROTL
        ADD X2
        ADD Y2
        STORE Z2
        HALT
X1      011111
X2      022222
Y1      033333
Y2      044444
Z1        0
Z2        0
```

65. Napiši program za odbijanje dva broja dvostruke točnosti!

Rješenje:

```
        *100
ODB     CLEARL
        CLEAR
        ADD Y1
        COMPL
        IAC
        ADD X1
        STORE Z1
        CLEAR
        ROTL
        STORE LINK
        CLEAR
        ADD Y2
        COMPL
        ADD X2
        ADD LINK
        STORE Z2
        HALT
X1      000000
X2      011111
Y1      000001
Y2      033333
Z1      0
Z2      0
LINK    0
```

66. Sadržaj lokacije 100_8 treba pomnožiti s 10_{10}. Koristiti postupke:
    a) množenja s bazom i pribrajanja
    b) uzastopnog zbrajanja. Ako se svaka instrukcija izvršava 2 mikrosekunde koji je postupak brži i za koliko ?

Rješenje:

a) množenje s bazom i pribrajanje

```
        *120
MULT10  CLEARL
        CLEAR
        ADD 100
        ROTL
        STORE T
        ROTL
        ROTL
        ADD T
        STORE 100
T        0
```

b) uzastopno zbrajanje

```
        *120
MU10    CLEAR
        ADD M10
        STORE COUNT
        CLEAR
        ADD 100
        ISZ COUNT
        JUMP .-2
        STORE 100
        HALT
COUNT    0
M10     177766
```

67. Napisati potprogram za kvadriranje broja. Broj se unosi u potprogram preko akumulatora (rezultat se takodjer nalazi u akumulatoru). Koji je najveći broj koji se može kvadrirati a da se ne pojavi greška (preliv)?

Rješenje:

```
          .
          .
          .
KVADRA    0
          STORE SPAS
          COMPL
          IAC
          STORE COUNT
          CLEAR
          ADD SPAS
          ISZ COUNT
          JUMP .-2
          JUMP I KVADRA
SPAS        0
COUNT       0
```

najveći broj je $\sqrt{0\ 111\ 111\ 111\ 111\ 111_2} = \sqrt{77777_8} =$

$= \sqrt{32768-1} = \sqrt{32767_{10}} = 181_{10}$

68. Potrebno je zamijeniti sadržaje dvije lokacije bez korištenja treće. Napiši program!

Rješenje:

```
          *50
                CLEAR
                ADD X
                ADD Y
                STORE X
                CLEAR
                ADD Y
                COMPL
                IAC
```

(nastavak na idućoj strani)

(nastavak)

```
        ADD X
        STORE Y
        CLEAR
        ADD Y
        COMPL
        IAC
        ADD X
        STORE X
        HALT
X       012345
Y       054321
```

69. Napisati potprogram koji će sadržaj akumulatora rotirati u lijevo za broj bitova naznačen u lokaciji iza instrukcije JMS !

Rješenje:

```
        *10000
POTP    0
        STORE POM
        CLEAR
        ADD I POTP
        COMPL
        IAC
        STORE COUNT
        CLEAR
        ADD POM
        ROTL
        ISZ COUNT
        JUMP .-2
        ISZ POTP
        JUMP I POTP
COUNT   0
POM     0
```

70. Pretpostavimo da se operacione instrukcije školskog računala izvršavaju 1 mikrosekundu, memorijske instrukcije 2 μsek. , a instrukcije s indirektnim adresiranjem 3 μsek. Potrebno je napisati potprogram koji će se izvršavati na školskom računalu. Izvodjenje potprograma treba trajati 100 milisekundi (vremensko kašnjenje). Uz svaku instrukciju potrebno je napisati vrijeme njezinog izvodjenja !

Rješenje:

```
KAS     0
        CLEAR       1 usek.
        ADD APOM    2 usek.
        STORE POM   2  "
        CLEAR       1  "
        ISZ POM     2  "
        JUMP .-2    1  "
        JUMP I KAS  2  "
APOM    060647
POM       0
```

100 msek = 100 000 μsek (100 000: 4 = $25\,000_{10}$)

```
 25000
-16384    2^14
  8616
- 8192    2^13
   424
-  256    2^8
   168
-  128    2^7
    40    2^5
          2^3
```

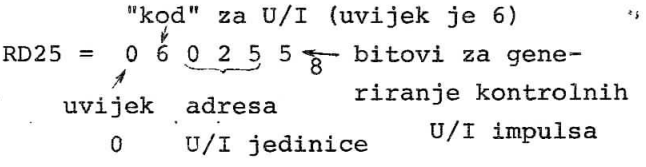
(petlja traje 4 μsek. Zadnja petlja traje 3 μsek, preostali dio potpr. traje 7 μsek.)

$25000_{10} = 060650_8$. Uz (APOM)=060647_8 potprogram će se izvršavati 100 002 μsek.

3. POVEZIVANJE ŠKOLSKOG RAČUNALA S ULAZNO/IZLAZNIM JEDINICAMA (Ulazno/izlazno programiranje)

Adresiranje ulazno/izlaznih jedinica kod "školskog računala" objašnjeno je u knjizi G.Smiljanić :"Osnove digitalnih računala".

Ulazno/izlazne instrukcije imaju slijedeći oblik:



Adresa U/I jedinice može biti simbolička (RDTTY) uz pretpostavku da je simbolička adresa prije toga negdje u programu definirana (npr. TTY= 1). U tom slučaju mašinski kod za instrukciju RDTTY je 060015_8 .
U ovim zadacima koristimo slijedeće instrukcije:

WR25 = 060256_8
RD30 = 060305_8
CL01 = 060014_8
STOP12 = 060123_8 /zaustavi U/I jedinicu 12_8
SKP112 = 061122_8
GO03 = 060031_8 /pokreni U/I jedinicu 3_8

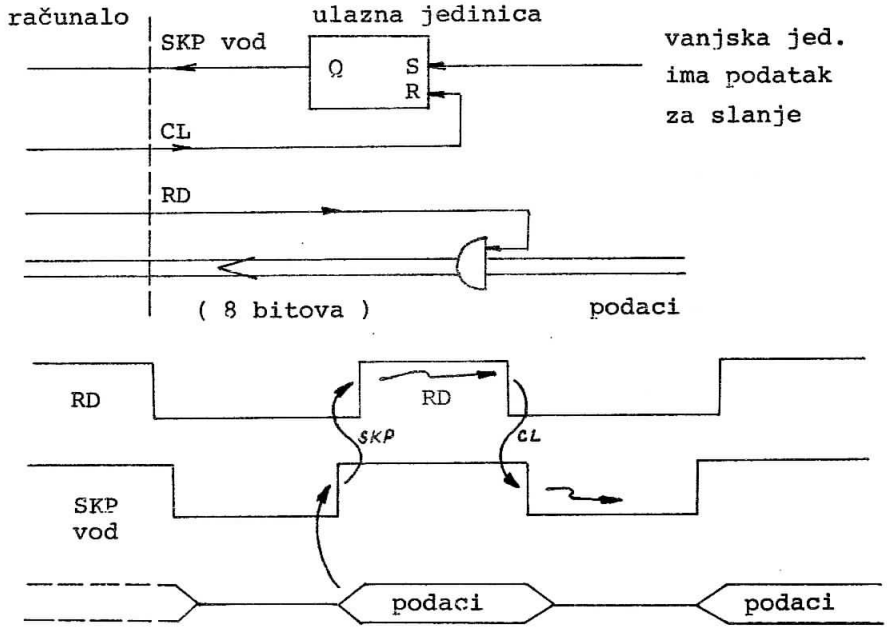
(ULAZNO / IZLAZNO PROGRAMIRANJE)

71. Napisati zatvoreni potprogram za prihvat podataka iz vanjske jedinice u računalo. Potprogram treba realizirati za programirani uvjetni prijenos podataka. Adresa vanjske jedinice računala je 3.

Rješenje:

```
ULAZ   0
       SKP3
       JUMP .-1
       CLEAR
       RD3
       CL3
       JUMP I ULAZ
```

72. Skicirati vremenski dijagram signala kod komunikacije računalo - ulazna vanjska jedinica (čitač papirne trake).

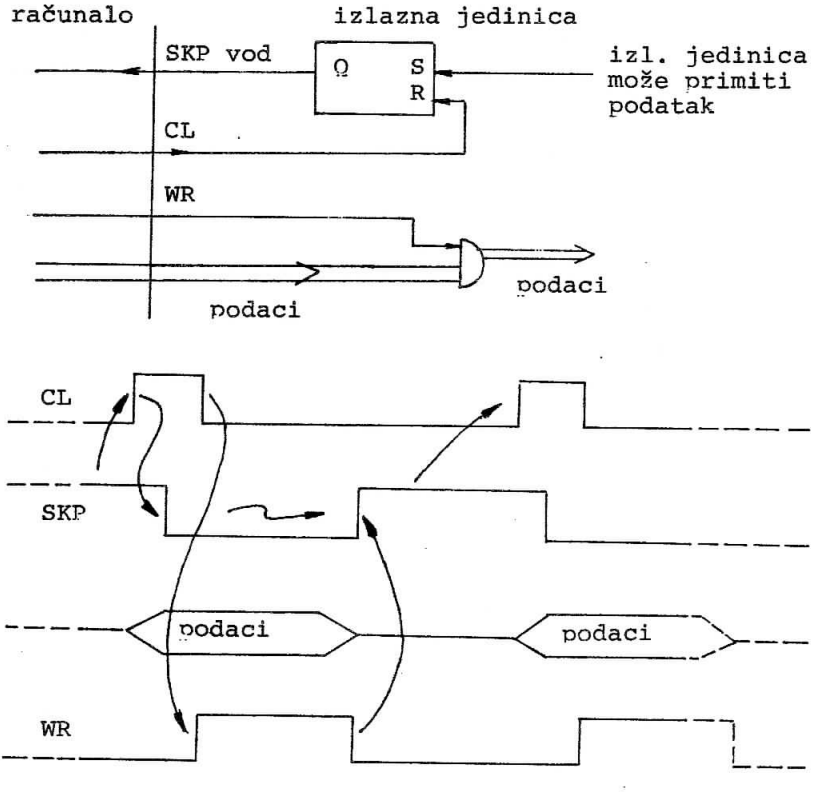


73. Napisati zatvoreni potprogram za slanje podataka iz računala u vanjsku jedinicu čija je adresa je 4. Koristiti programirani uvjetni prijenos podataka.

Rješenje:

```
IZLAZ    0
         SKP4
         JUMP .-1
         CL4
         WR4
         JUMP I IZLAZ
```

74. Skicirati vremenski dijagram signala kod komunikacije računalo - izlazna jedinica (bušač papirne trake).



75. Potprogram "ZVONO" potrebno je nadopuniti instrukcijama koje nedostaju. Pozivom potprograma aktivira se 5 puta zvono na teleprinteru.

```
ZVONO    0
         CLEAR
         ADD M5
         STORE COUNT
         CLEAR
         ADD K207
         JMS  IZLAZ     /ispis znaka
         JMS  KAS       /vremensko kašnjenje
         ISZ COUNT       kako bi se zvono čulo
         JUMP .-3
         JUMP I ZVONO
M5       177773         / -5
K207     000207         / ASCII "kod" za zvono
COUNT    0
```

76. Upotpuni zadani program koji treba štampati binarni sadržaj lokacije AC, bit po bit (od lijeva na desno). Za ispitivanje binarne vrijednosti svakog od 16_{10} binarnih mjesta koristi link. (ASCII kod za "0"=260_8 , a za "1" =261_8). Adresa štampača je 33_8.

Rješenje:

```
         *20
PO       CLEAR
         CLEARL
         ADD AC
         ROTL
         STORE AC
         CLEAR
         ADD ASC
         SKIPL
         JUMP.+2
         IAC
```

(nastavak na drugoj strani)

```
                (nastavak)
                    ↓
                WR33
                ISZ COUNT
                JUMP PO
                HALT
ASC             000260
COUNT           177760
AC              010101
```

77. Analizirati program za prijem znakova s vanjske jedinice 03. Ako program ima instrukcije koje onemogućavaju takvu komunikaciju treba ih precrtati !

```
*1000
START   CLEAR
        CL03
         .
         .
         .
        JMS ULAZ1
        STORE MEM
         .
         .
         .
        HALT
ULAZ1     0
        SKP03
        JUMP .-1
        ~~CL03~~
        CLEAR
        RD03
        CL03
        ~~ISZ ULAZ1~~
        JUMP I ULAZ1
```

78. Analizirati potprogram za predaju znakova vanjskoj jedinici 33. Ako potprogram sadrži instrukcije koje onemogućavaju programirani uvjetni prijenos podataka u vanjsku jedinicu treba ih prekrižiti !

```
OUT     0
        SKP33
        JUMP .-1
        CL33
        CLEAR
        WR33
        JUMP I OUT
```

79. Upiši potreban sadržaj u lokaciju M da sa teleprintera budu ispravno primani oktalni brojevi u ASCII kodu i pretvarani u binarni broj. (ASCII kod za "0" je 260_8).

a)

```
        TTY=1
          .
          .
        SKPTTY
        JUMP .-1
        CLEAR
        RDTTY
        CLTTY
        AND M
        STORE BROJ
        HALT
M       000007  <--
BROJ    0
```

b)

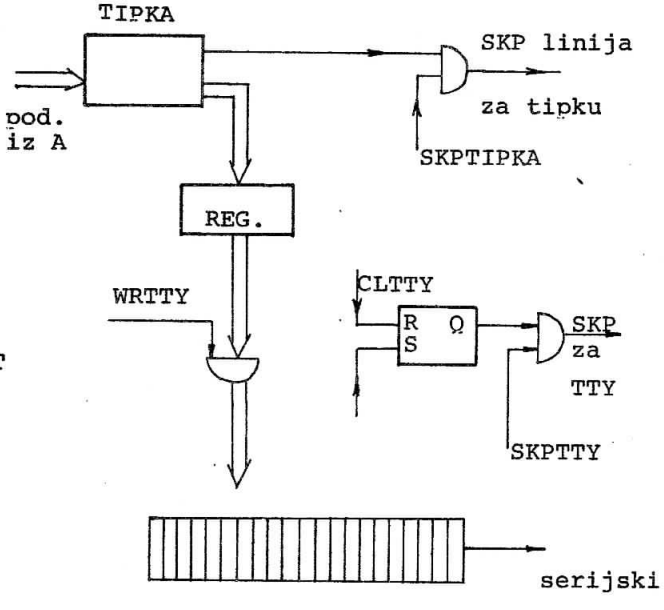
```
          .
          .
        SKPTTY
        JUMP .-1
        CLEAR
```

(nastavak na idućoj strani)

```
(nastavak)
    ↓
    RDTTY
    CLTTY
    ADD M
    STORE BROJ
M   177520        / -260₈   ←
BROJ  0
```

80. Upotpuni zadani program tako da štampa sadržaje 100_8 lokacija iz tabele podataka s početnom adresom 2000_8, ali pod slijedećim uvjetom: pored teleprintera na računalo je spojena tipka, a tek pritiskom na tipku, podaci se prenose iz računala na teleprinter. Ako se tipka pritisne kratkotrajno, tada se štampa samo jedan podatak.

```
TTY    =1
TIPKA  =2
       .
       .
       .
       CLEAR
       CLTTY
       WRTTY
       ADD A2000
       STORE POX
       CLEAR
       ADD M100
       STORE COUNT
B      SKPTIPKA
       JUMP .-1
       SKPTTY
       JUMP .-1
       CLTTY
       CLEAR
       ADD I POX
       ↓
```



(nastavak na idućoj strani)

(nastavak)

```
        WRTTY
        ISZ POX
        ISZ COUNT
        JUMP B
        HALT
A2000   002000
POX       0
M100    177634
COUNT     0
```

81. Napiši zatvoreni potprogram koji će na izlaznom formularu otisnuti "NASLOV:"

Rješenje:

```
        *23
NAS       0
          CLEAR
          ADD MB
          STORE COUNT
          CLEAR
GO        ADD AN
          JMS IZLAZ
          CLEAR
          ISZ GO
          ISZ COUNT
          JUMP GO
          ADD  P7
          ADD GO
          STORE GO
          CLEAR
          JUMP I NAS
```

(nastavak na idućoj strani)

(nastavak)

```
AN   000316
AA   000301
AS   000323
AL   000314
AO   000317      / ASCII kod za:
AV   000326        "NASLOV:"
A    000272
MB   177771
COUNT  0
P7   000007
```

(PREKID PROGRAMA)

82. Potrebno je napisati program koji će štampati poruku "ZDRAVO!" kada se aktivira bilo koja tipka tastature teleprintera, koristeći prekidni prijenos. U medjuvremenu se izvodi neki program. (U slučaju zahtjeva za prekid (PC) se sprema u lokaciju 0, a izvodi se instrukcija koja se nalazi u lokaciji 1).

Rješenje:

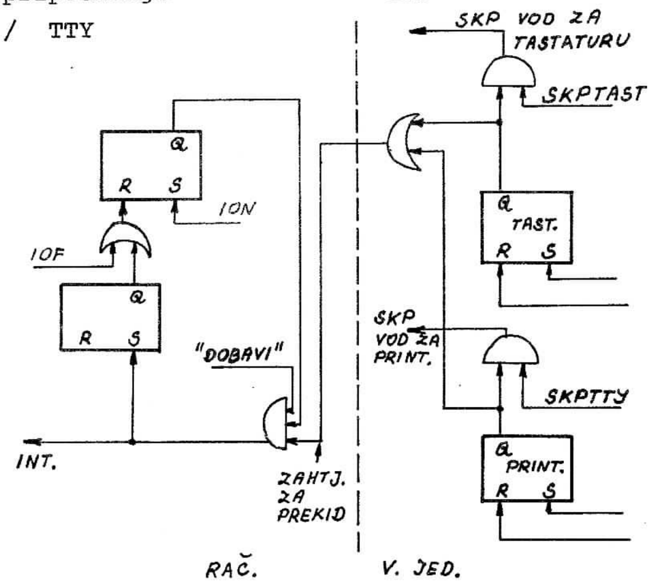
```
                    ( PC )

*0
0                     0
1                   JUMP SERVIS
                      .
                      .
                      .
*200
200                 CLEAR
                    CLEARL
                    ADD BROJ
                    STORE BRLO
                    CLEAR
                    ADD ABLOK
                    STORE BLOK
```

(nastavak na idućoj strani)

```
        (nastavak)
           ↓
        CLEAR
        ION
         .
         .
         .
        glavni program
         .
         .
        HALT
SERVIS  STORE SPASI
        CLEAR
        ROTL
        STORE LINK
        CLEAR
        SKPTAST
        SKIP
        JUMP TASTA
        SKPTTY
        SKIP
        JUMP TTYP
        JUMP NATRAG
TASTA   CLTAST
        WRTTY       /pripremanje
        JUMP NATRAG /  TTY
TTYP    CLTTY
        ADD I BLOK
        WRTTY
        CLEAR
        ISZ BLOK
        ISZ BRLO
        JUMP NATRAG
        HALT
         ↓
         *
```

```
            *
            ↓
NATRAG   ADD LINK
         ROTR
         ADD SPASI
         ION
         JUMP I 0
SPASI      0
LINK       0
BROJ     177771
BRLO       0
ABLOK    000400
BLOK       0
           .
           .
           .
*400
400      000332   /Z
         000304   /D
         000322   /R
         000301   /A
         000326   /V
         000317   /O
         000241   /!
         END
```



83. Programiranim uvjetnim prijenosom prenosi se na liniju (adresa 10_8) , 100_8 znakova iz polja mem. s početnom adresom 2000_8, do zaustavljanja programa. Poseban elektronički sklop nadzire ispravnost znakova na liniji. U slučaju neispravnosti , traži odmah ponavljanje znaka dajući zahtjev za prekid programa. Napiši prekidni potprogram da se znak ponovi !

```
*100
POCET   CLEAR
        ADD M100
        STORE COUNT
        CLEAR
        ADD A2000
        STORE PO
        ION
OUT     ISZ PO
        ISZ COUNT
        SKIP
        JUMP HALT
        CLEAR
        ADD I PO
        WR10
        SKP10
        JUMP .-1
        JUMP OUT
HALT    HALT
COUNT     0
M100    177677      / -101
PO        0
A2000   001777
```

Rješenje:

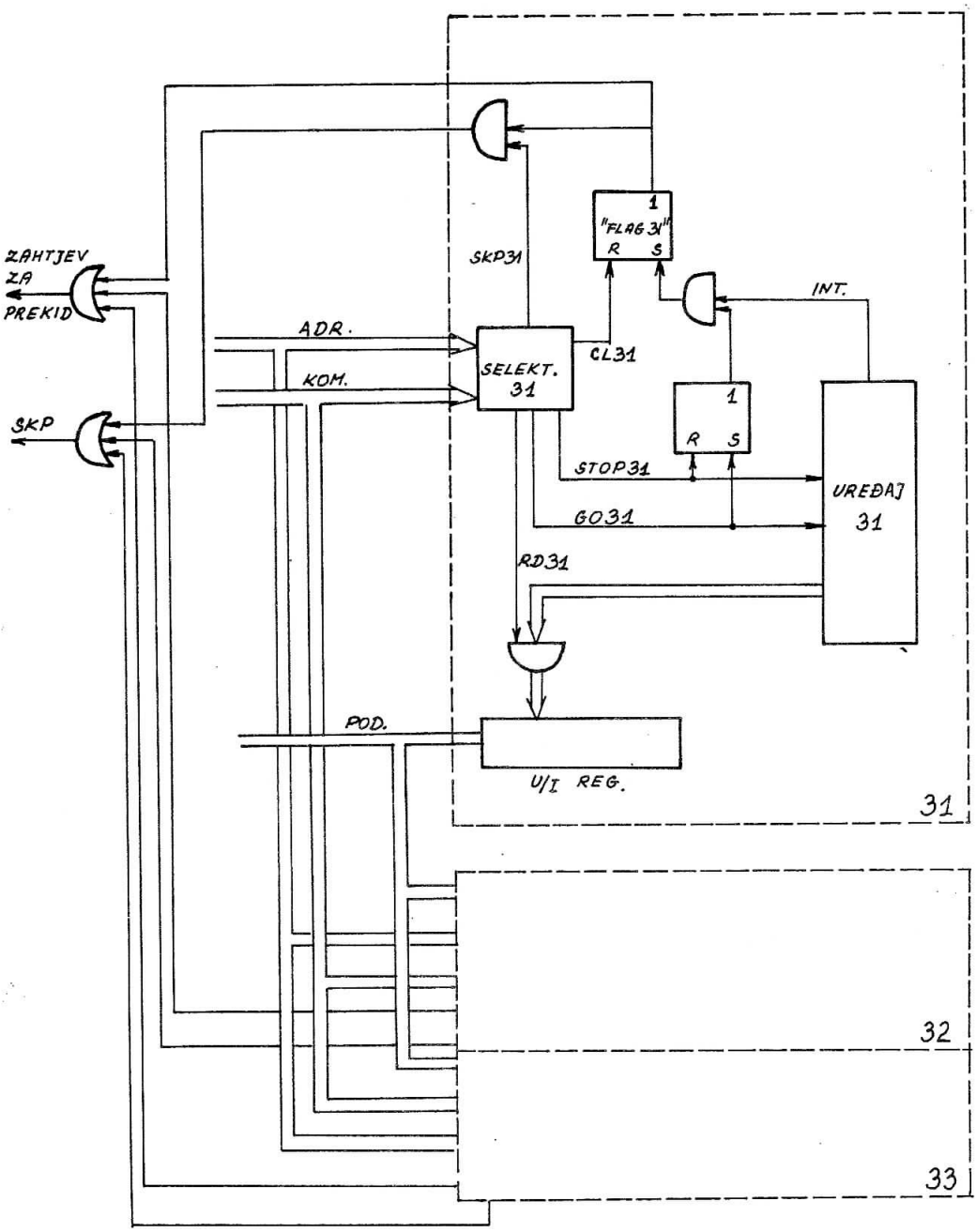
```
*0
0       0
1       CLEAR
        COMPL
        ADD PO
        STORE PO
        CLEAR
        COMPL
        ADD COUNT
        STORE COUNT
        CLEAR
        CL10
        ION
        JUMP I 0
        END
```

84. Programiranim uvjetnim prijenosom treba prenijeti 50_8 podataka iz vanjske jedinice 32_8 u blok memorije s početnom adresom 700_8. Ako tokom prijenosa tih podataka vanjska jedinica 33 zatraži prekid programa, ponovi unošenje svih podataka od početka bloka memorije- 700_8 . Prekidni potprogram počinje od lokacije 0 i preko njega se unosi zahtjev za ponavljanjem prijenosa. Napiši program koji omogućava takvu komunikaciju !

85. Tri vanjske jedinice 31, 32 i 33 spojene su na prekidnu liniju računala. Svaki uredjaj (vanjska jedinica) sprema podatke u svoj blok memorije. Nakon što pojedine jedinice pošalju 100_{10} podataka, treba ih isključiti! Realizirati problem a) hardverski (sklopovski) i b) programski. Naredba "GO" uključuje vanjsku jedinicu, a naredba "STOP" je zaustavlja .

Rješenje:

a)


ZAHTJEV ZA PREKID
SKP
ADR.
KOM.
SKP31
SELEKT. 31
1
"FLAG 31"
R S
CL31
INT.
1
R S
STOP31
GO31
RD31
UREĐAJ 31
POD.
U/I REG.
31
32
33

b)

```
        *0
        0       0
        1       JUMP  ULAZ
                 .
                 .
                 .
        200     CLEAR
                ADD M100
                STORE BROJ1
                STORE BROJ2
                STORE BROJ3
                CLEAR
                ADD AB1
                STORE POB1
                CLEAR
                ADD  AB2
                STORE POB2
                CLEAR
                ADD AB3
                STORE POB3
                CLEAR
                GO31
                GO32
                GO33
                ION
                 .
                 .
                 gl. program
                 .
                HALT
M100            177634  /-100(10)
BROJ1           0
BROJ2           0
BROJ3           0
AB1             001000  / B1
AB2             001100  / B2
AB3             001200  / B3
POB1            0
POB2            0
POB3            0
AK              0
LINK            0
```

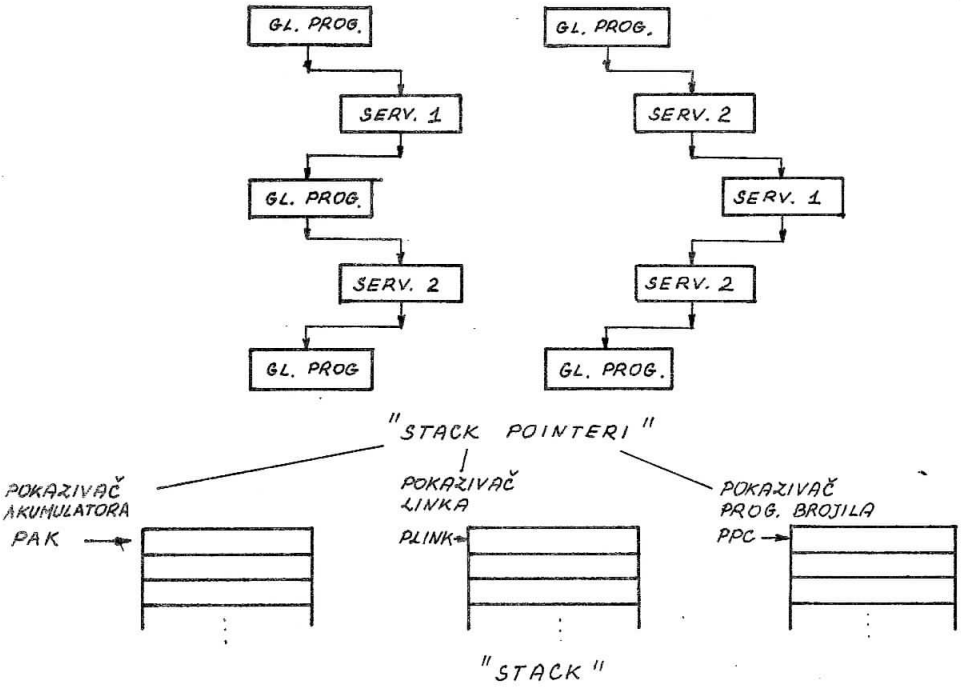
↓ *

↓ *

```
ULAZ    STORE AK
        CLEAR
        ROTL
        STORE LINK
        SKP31
        SKIP
        JUMP ULAZ1
        SKP32
        SKIP
        JUMP ULAZ2
        SKP33
        SKIP
        JUMP ULAZ3
IZLAZ   CLEAR
        ADD LINK
        ROTR
        ADD AK
        ION
        JUMP I 0
ULAZ1   CLEAR
        RD31
        CL31
        STORE I POB1
        ISZ POB1
        ISZ BROJ1
        SKIP
        STOP31
        JUMP IZLAZ
ULAZ2   CLEAR
        RD32
        CL32
        STORE I POB2
        ISZ POB2
        ISZ BROJ2
        SKIP
        STOP32
        JUMP IZLAZ
ULAZ3   CLEAR
         .
         .
        itd.
```

86. Računalo poslužuje jedinice 01 i 02 od kojih je jedinica 02 izrazito brža. Neka se omogući da se za vrijeme posluživanja sporije jedinice obradi primljeni zahtjev za prekid od brže jedinice ! (Koristiti "STACK POINTER")

Rješenje:




```
*0
0          0
1          JUMP SERV
           .
           .
```

(nastavak na idućoj strani)

↓ +

(nastavak)

```
        ↓+
        .
        .
POCET   CLEAR
        ADD APAK
        STORE PAK
        CLEAR
        ADD APLINK
        STORE PLINK
        CLEAR
        ADD APPC
        STORE PPC
        CLEAR
        ION
        .
        .
        gl. program
        .
        .
        HALT
SERV    STORE I PAK
        CLEAR
        ROTL
        STORE I PLINK
        CLEAR
        ADD 0
        STORE I PPC
        CLEAR
        ISZ PAK
        ISZ PLINK
        ISZ PPC
        SKP01
        SKIP
        JUMP SERV1
        SKP02
        SKIP
        JUMP SERV2
        HALT
        ↓*
```

```
        ↓*
SERV1   ┌──────────┐
        │          │
        │          │
        │          │
        │          │
        │          │
        │          │
        │          │
        JUMP IZLAZ
SERV2   ION
        ┌──────────┐
        │          │
        │          │
        │          │
        │          │
        │          │
        JUMP IZLAZ
IZLAZ   IOF
        CLEAR
        COMPL
        ADD PAK
        STORE PAK
        CLEAR
        COMPL
        ADD PLINK
        STORE PLINK
        CLEAR
        COMPL
        ADD PPC
        STORE PPC
        CLEAR
        ADD I PPC
        STORE 0
        CLEAR
        ADD I PLINK
        ROTR
        ADD I PAK
        ION
        JUMP I 0
```

(nastavak na idućoj strani)

↓o

(nastavak)

```
                          ↓0
                APLINK    000500  /pokazivač za L
"STACK          APAK      000520  /pokazivač za  A
 POINTERI"      APPC      000540  /pokazivač za  PC
                PLINK       0
                PAK         0
                PPC         0
                            .
                            .
                            .
                            .
```

87. Školsko računalo izvršava program čije izvodjenje traje 1 sat. Kad se prekine napajanje računala sadržaji memorijskih lokacija ostaju sačuvani medjutim sadržaji registara se izbrišu. Uz računalo postoji poseban sklop koji na vrijeme otkriva smanjenje napona i izaziva bezuvjetni zahtjev za prekid sa skokom na lokaciju 0 u memoriji računala. Potrebno je napisati program za obradu prekida koji će pospremiti sadržaje bitnih registara u trenutku nestanka električne energije i zaustaviti izvodjenje programa. Takodjer je potrebno napisati program koji će omogućiti nastavak izvodjenja posla od mjesta prekida (kada dodje električna energija). Izvodjenje tog programa aktivira operater ručno!

Rješenje:

```
*0                          *20        /inicijalizacija
0       0                   INIC   CLEAR
      STORE AK                     ADD LINK
      CLEAR                        ROTR
      ROTL                         CLEAR
      STORE LINK                   ADD AK
      HALT                         JUMP I 0
AK     0
LINK   0
```

(DMA prijenos)

88. Kod DMA komunikacije izmedju računala i vanjske jedinice (i obratno) koristi se direktni pristup u memoriju (bez posredstva akumulatora i bez posebnih instrukcija za prijenos svakog podatka). Time se omogućava procesoru da za vrijeme prijenosa može obavljati neki drugi zadatak. Koji osnovni hardverski elementi moraju postojati za tu vrstu komunikacije ?

Rješenje:

Vanjska jedinica:
- status bistabil
- adresni registar
- brojilo
- registar podataka
- upravljačka logika

Računalo :
- upravljačka logika
- DMA adresni registar

89. Koji su osnovni načini DMA prijenosa ?

Rješenje:

Tri najčešće upotrebljavana DMA načina prijenosa su:

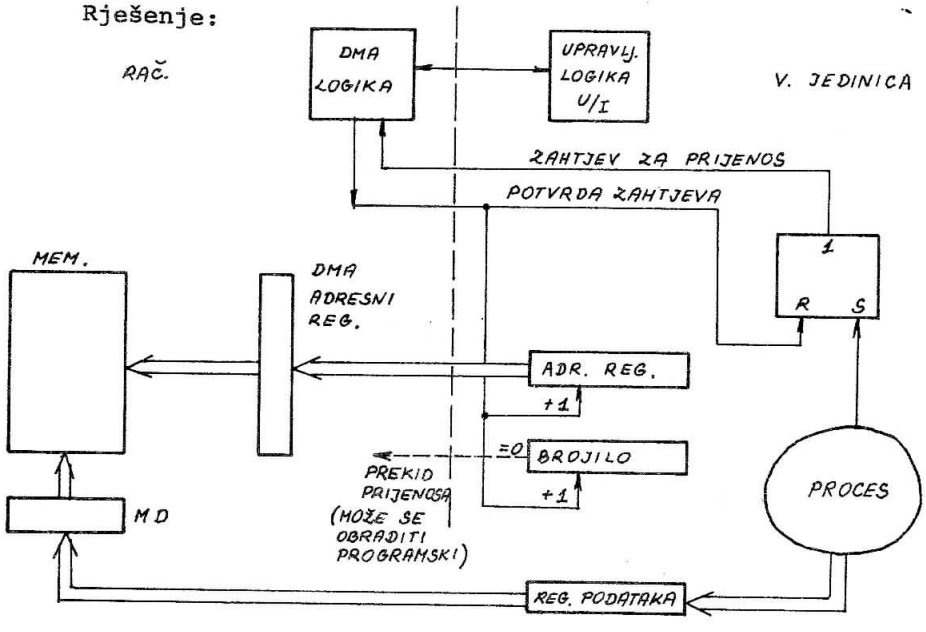
a) zaustavljanje procesora
   - prijenos podataka direktno u memoriju ili iz memorije
   - prednost , jednostavan hardver
   - nedostatak , procesor se zaustavlja
   (na taj način se može prenijeti praktički neograničen broj podataka)

b) kradja ciklusa ("Cycle Stealing")
- prednost , procesor se ne zaustavlja već radi svoj posao, a DMA prijenos ga samo malo usporava na taj način da se uzima jedan ciklus instrukcije za prijenos jednog podatka
- nedostatak , složeniji hardver

c) koristi se svojstvo da se adresiranje memorije vrši samo u odredjenim trenucima izvodjenja instrukcije, a u ostalo vrijeme je adresna logika slobodna !
- prednost , to je najbrži DMA prijenos , ne usporava rad procesora niti ga zaustavlja
- nedostatak , hardver je još kompliciraniji , potrebne su brze memorije
(kod nekih mikroprocesora se na taj način uz mem. ciklus od 560 nsek postiže prijenos jednog podatka svakih 1,2 μsek.).

90. Skicirati osnovne elemente za DMA prijenos kradjom ciklusa.

Rješenje:



91. Rezultate mjerenja vanjske jedinice računala treba unijeti u blok memorije računala. Veličina bloka je 100_8 lokacija, a početna adresa 3000_8 . Nakon što se unesu svi podaci, treba isključiti uredjaj. Napisati program i skicirati hardversku organizaciju:

a) koji će pripremiti direktno unošenje podataka u memoriju, aktivirati prijenos (mjerenje) i čekati dok se svi podaci ne prenesu.

b) kao pod a) ali tako da se "kradja ciklusa" obavlja tokom izvodjenja nekog glavnog programa.

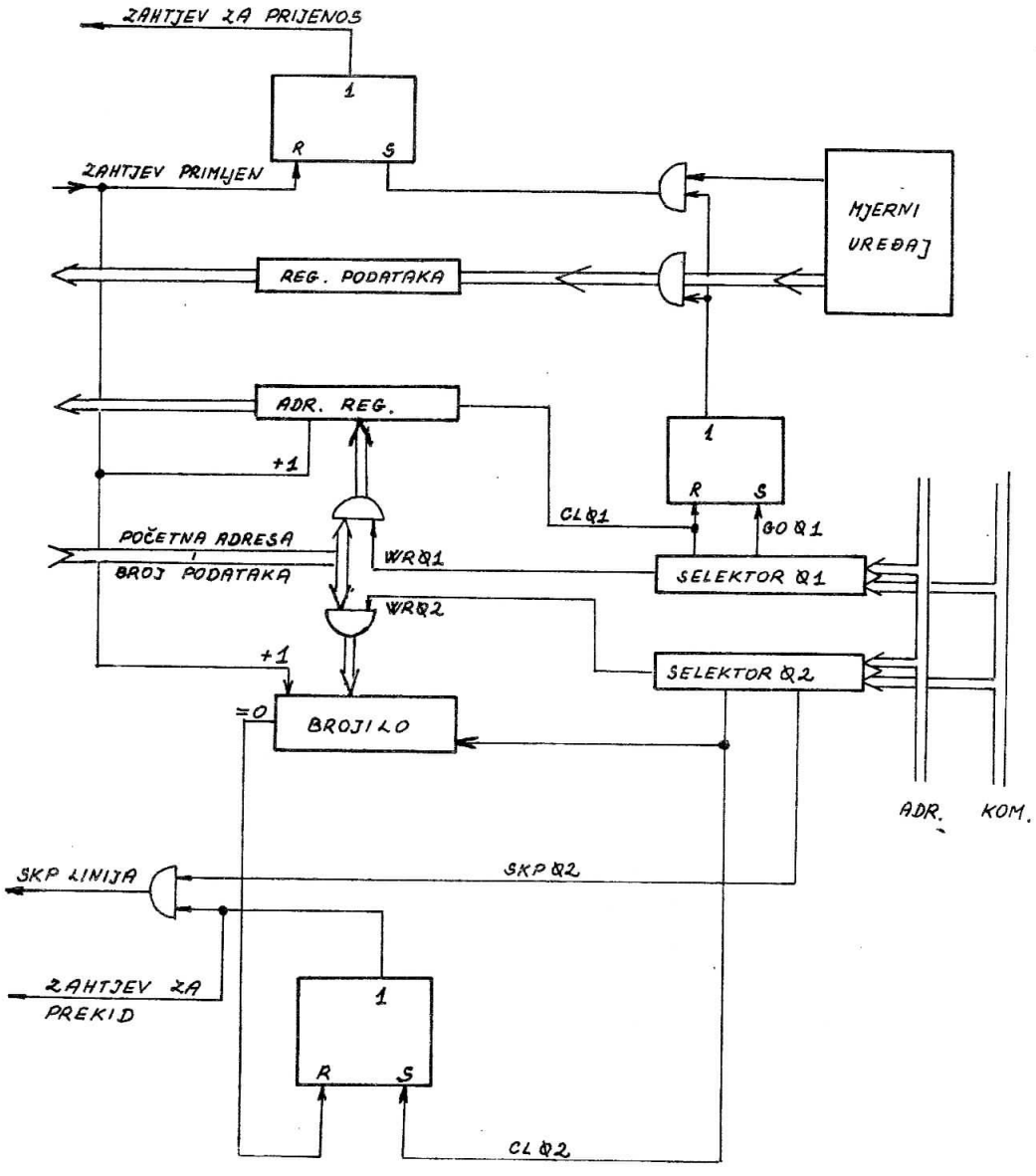
c) koji traženi prijenos obavlja "programiranim prijenosom"

d) koji će traženi prijenos obaviti "prekidnim prijenosom".

Rješenje:

a)

```
                .          program je predvidjen za
                .          hardversku organizaciju na
            CL01           slijedećoj strani !
            CL02
            CLEAR
            ADD ABLOK          inicijalizacija DMA
            WR01
            CLEAR
            ADD M100
            WR02
            CLEAR
            GO01               start DMA
            SKP02
            JUMP .-1
            CL01               završetak DMA
            CL02
              .
              .
            HALT
ABLOK       002777     / potrebno je predvidjeti
M100        177677       da će prije prvog podatka
                         impuls "primljen zahtjev"
                         uvećati za 1 adr. reg.i brojilo!
```



Slika medjusklopa za DMA prijenos

b)

```
                0       0
                1       CL01
                        CL02
                        JUMP I 0
                         .
                         .
                         .
                100     CL01
                        CL02
                        CLEAR
                        ADD ABLOK
                        WR01
                        CLEAR
                        ADD M100
                        WR02
                        CLEAR
                        GO01    / start DMA
                        ION     / omogući prekid
                         .
                         .
                        gl. program
                         .
                         .
                        HALT
              ABLOK     002777
              M100      177677
                         .
                         .
```

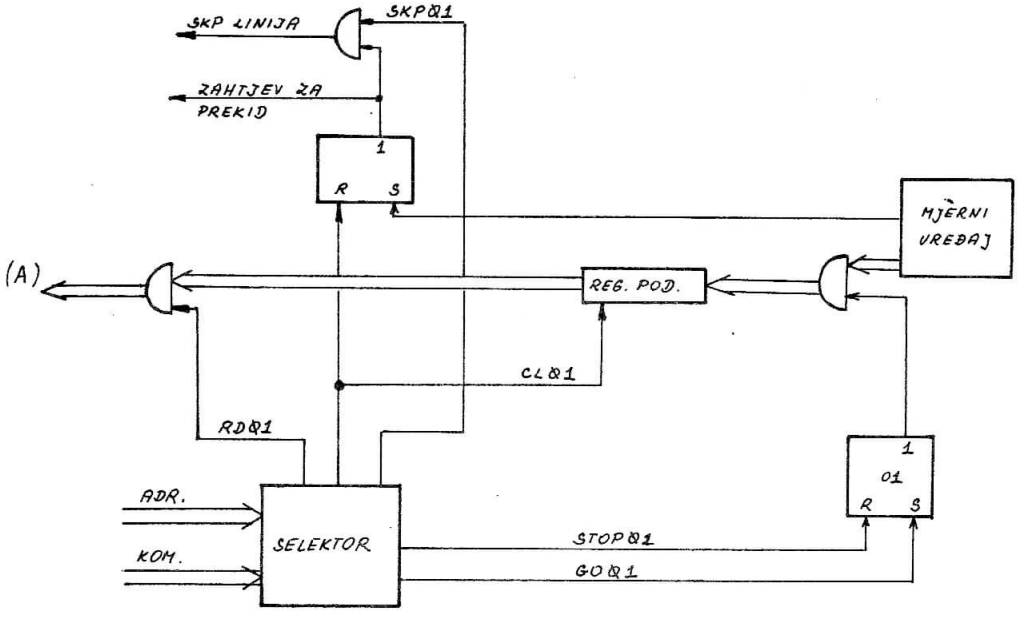
c) "programirani prijenos"

```
                         .
                         .
              *600
              START     CL01
                        CLEAR
                        ADD ABLOK
                        STORE POBLOK
                        CLEAR
                        ADD M100
                        STORE BROJLO
                        GO01
```

(nastavak na idućoj strani)

(nastavak)

```
                    ↓
PONOVO      SKP01
            JUMP .-1
            CLEAR
            RD01
            CL01
            STORE I POBLOK
            ISZ POBLOK
            ISZ BROJLO
            JUMP PONOVO
            STOP01
            HALT
POBLOK       0
ABLOK       003000
M100        177700
BROJLO       0
```



Skica medjusklopa za "programirani prijenos" i "prekidni prijenos"

d) prekidni prijenos

```
        *0
        0       0
        1       STORE AK
                CLEAR
                RD01
                CL01
                STORE I POBLOK
                ISZ POBLOK
                CLEAR
                ADD AK
                ISZ BROJLO
                ION
                JUMP I 0
        ABLOK   003000
        POBLOK    0
        M100    177700
        AK        0
        LINK      0
        BROJLO    0
        START   CL01      / početak programa
                CLEAR
                ADD ABLOK
                STORE POBLOK
                CLEAR
                ADD M100
                STORE BROJLO
                GO01
                ION
                 .
                 .
                gl. program
                 .
                 .
                HALT
```

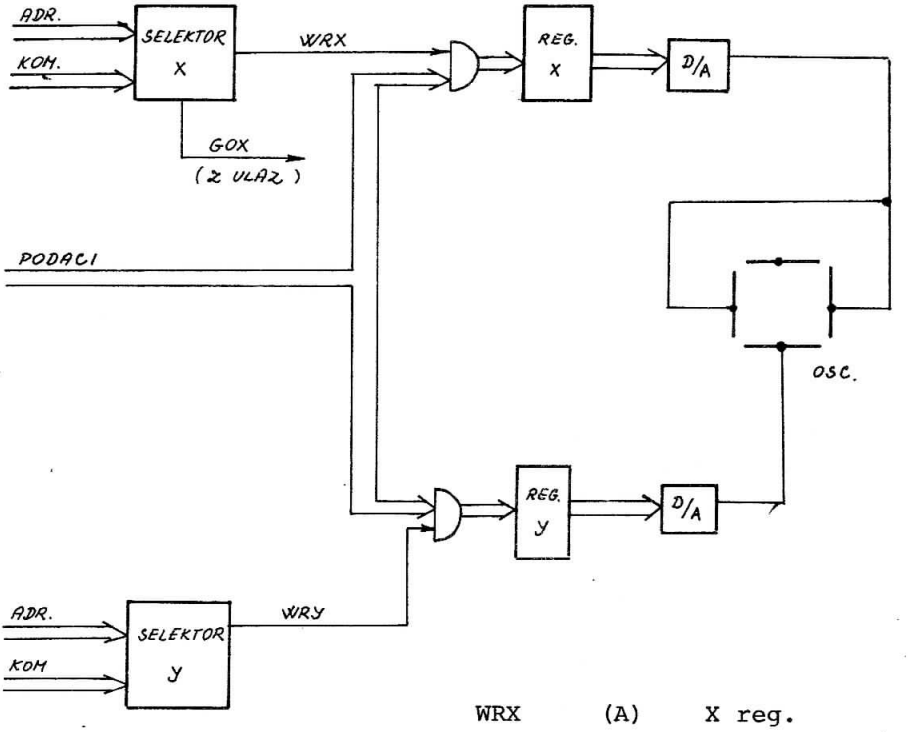
92. Vanjski uredjaj 34 prenosi izravno u memoriju (DMA prijenos) podatke počevši od lokacije 3000_8. Ako u toku prijenosa vanjski uredjaj 35 postavi signal, izravni prijenos treba nastaviti od lokacije 2000_8 . Osnovni program neprestano ispituje signal vanjskog uredjaja 35 da bi ustanovio postavljanje signala i usmjerio daljnji tok izravnog prijenosa. Potrebno je nadopuniti taj dio programa !

```
                .
                .
            CL34
            CL35
            CLEAR
            ADD A3000
            WR34
            GO34
            GO35
            SKP35
            JUMP .-1
            CL34
            ------
            CLEAR
            ADD A2000
            WR34
            GO34
            ------
                .
                .
                .
            HALT
A3000       003000
A2000       002000
```

(OSCILOSKOPSKI PRIKAZ PODATAKA -

bezuvjetni programirani prijenos)

94. Skicirati osnovne elemente koji su potrebni za osciloskopski prikaz podataka.



WRX	(A)	X reg.
WRY	(A)	Y reg.
GOX		osvijetli !

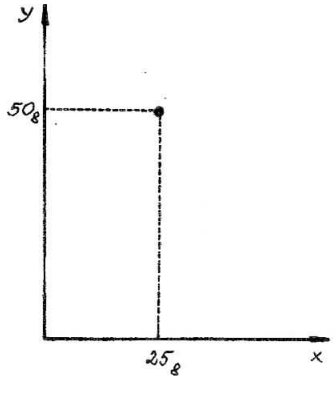
(pretpostavimo da je u slijedećim zadacima:
$X=010_8$, $Y=011_8$ dakle WRX = 060106, WRY= 060116,
$GOX=060101_8$)

95. Prikazati na ekranu osciloskopa točku s koordinatama $X= 25_8$ i $Y= 50_8$.

Rješenje:

```
*100
POC CLEAR
    ADD K25
    WRX
    CLEAR
    ADD K50
    WRY
    GOX
     .
     .
     .
    HALT
K25 000025
K50 000050
```

(kod rješavanja ovih zadataka pretpostavljeno je da se D/A pretvorba izvrši za vrijeme izvršavanja jedne instrukcije. Ako taj uvjet nije udovoljen potrebno je nakon inicijalizacije D/A pretvorbe ostvariti vremensko kašnjenje (programski) .

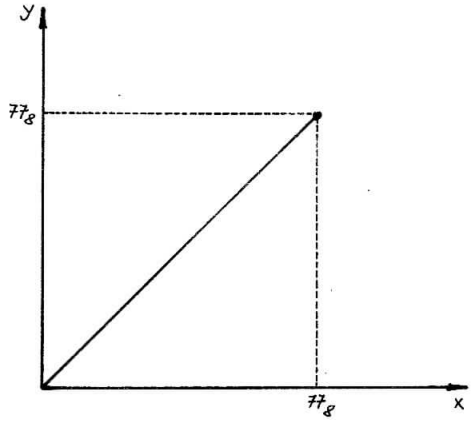


96. Prikazati na ekranu osciloskopa pravac nagiba $tg\alpha = 1$.

Rješenje:

```
        *500
PONOVO  CLEAR
        ADD X
        COMPL
        STORE BROJLO
        CLEAR
PETLJA  IAC
        WRX
        WRY
        GOX
        ISZ BROJLO
        JUMP PETLJA
        JUMP PONOVO
X       000077
BROJLO    0
```

(treba razlikovati "ADD X" - X je simbolička adresa memorijske lokacije i"WRX"- X je adresa vanjske jedinice (registra za prikaz horizontalnog otklona))!

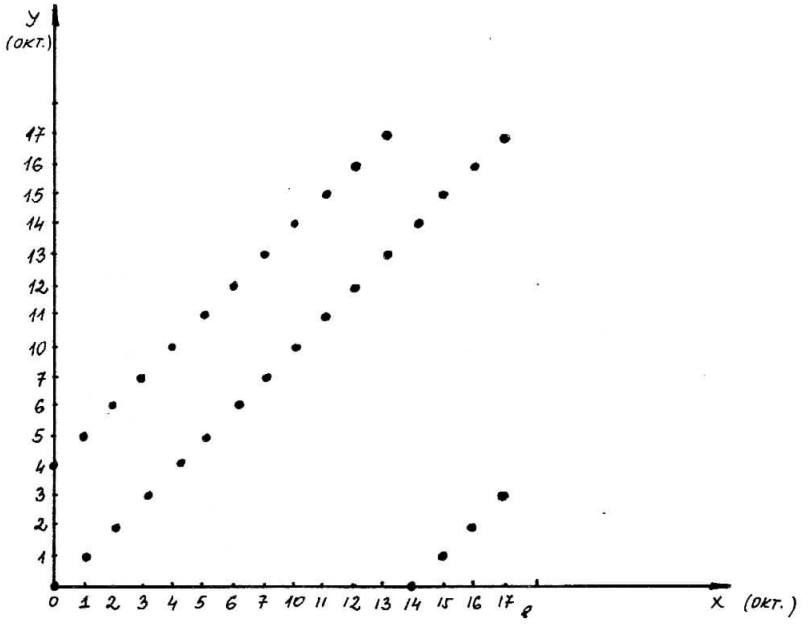


97. Analiziraj zadani program osciloskopskog prikaza i nacrtaj sliku. Instrukcija "GOX" označava trenutak iscrtavanja koordinata na ekranu osciloskopa !

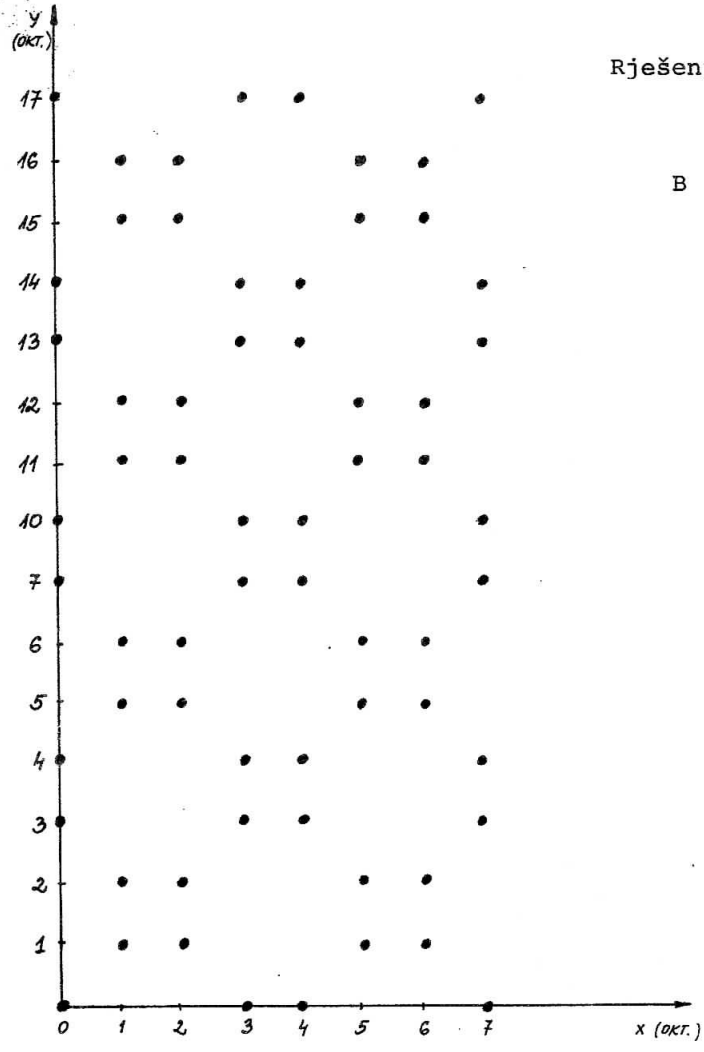
```
        *50
B       GOX
        COMPL
        AND K7
        WRY
        GOX
        IAC
        AND K7
        WRX
        ADD M4
        AND K7
        WRY
        JUMP B
M4      177774      /-4
K7      000007
```

98. Analiziraj program osciloskopskog prikaza i nacrtaj generiranu sliku . Za svaki pravac upiši koordinate krajnjih točaka na ekranu.

```
        *0
BEG     CLEAR
        ADD Z
        AND MSK17
        WRX
        WRY
        GOX
        ADD P4
        AND MSK17
        WRY
        GOX
        ISZ Z
        JUMP BEG
        JUMP BEG
MSK17   000017
P4      000004
Z         0
```



99. Napiši program koji će na ekranu osciloskopa iscrtavati slijedeću sliku:



Rješenje:

```
B     CLEAR
      ADD Z
      AND M17
      WRY
      CLEAR
      ADD Z
      AND M7
      WRX
      GOX
      COMPL
      AND M7
      WRX
      GOX
      ADD K4
      AND M7
      WRX
      GOX
      COMPL
      AND M7
      WRX
      GOX
      ISZ Z
      JUMP B
      JUMP B
M7    000007
M17   000017
K4    000004
Z       0
```

4. RJEŠAVANJE PROBLEMA KOD RAČUNALA SA ZADANIM KARAKTERISTIKAMA

4.1 HIPOTETSKO RAČUNALO

Ako u zadatku nije posebno objašnjeno računalo onda vrijedi:

- instrukcioni set i kodovi kao kod školskog računala
- 12 bitna riječ
- memorijsko adresiranje kao kod miniračunala PDP-8
- svi brojevi su oktalno pisani

100. Napiši u binarnom obliku slijedeće naredbe :
("kod" za STORE je 3_8)

		Rješenje (binarno):
a)	177 STORE 100	011 001 000 000
	377 STORE 100	011 001 000 000
	377 STORE 350	011 011 101 000

("kod" za AND je 0)

b)	520 AND 172	000 001 111 010
	522 AND 572	000 011 111 010
	622 AND 672	000 010 111 010

("kod" za JUMP je 5)

150 JUMP 0	101 000 000 000
450 JUMP 400	101 010 000 000
2000 JUMP 2100	101 011 000 000
150 JUMP I 0	101 100 000 000

101. Analiziraj mali potprogram i odredi koje će tri lokacije memorije biti promijenjene i koliki će biti njihov sadržaj nakon što potprogram obidje petlju tri puta. Početna vrijednost tih lokacija je 0.

```
*500      :
500  ISZ 177
     ISZ 500
     JUMP 500
        .
        .
```

Rješenje:

Adresa mem. lokacije	sadržaj
0177	0001
0400	0001
0401	0001

102. Analizirati program i odrediti u koje će lokacije biti upisana 0 nakon što se program zaustavi! (Neka je kod za HALT = 0000). Program počinje od lokacije 500_8 .

```
      .
      .
500   CLEAR
501   STORE 100
502   ISZ 501
503   JUMP 501
      .
      .
```

Rješenje:

lokacije 100 - 177
i 400 - 501

103. Analizirati program i odrediti u koje će memorijske lokacije biti upisan sadržaj lokacije 505. Napisati koje su to lokacije i njihov sadržaj !

```
      .
      .
476   CLEAR
      ADD 505
      STORE 175
      ISZ 500
      ISZ 505
      JUMP 476
      HALT
      7770
      .
      .
```

Rješenje:

lok.	sadržaj
175	7770
176	7771
177	7772
400	7773
401	7774
402	7775
403	7776
404	7777

104. Analiziraj zadani program i odredi koje će lokacije biti promijenjene nakon izvodjenja programa. Program počinje od lokacije 1101_8 . (Računalo ima organizaciju kao PDP-8, a instrukcioni set školskog računala).

```
                .
                .
1100    7776
1101    CLEAR
        STORE 1000
        COMPL
        ADD 1102
        STORE 1102
        ISZ 1100
        JUMP 1101
        HALT
```

Rješenje: lokacije koje će biti promijenjene (oktalno):

adresa	novi sadržaj
1000	0000
1100	0000
1102	3176
177	0000

105. Analiziraj zadani program i odredi u koje će lokacije biti upisana 0 nakon izvršenja programa ? Program počinje od lokacije 1101_8 . Računalo je 12 bitno, adresiranje je kao kod PDP-8, a instrukcije kao kod školskog računala. Kodovi za instrukcije su: CLEAR = 7200_8 , STORE = 3_8 , COMPL = 7040_8 , ADD = 1_8 , ISZ = 2_8 , JUMP = 5_8 , HALT = 7402_8). Napisati uz zadani simbolički program mašinski kod programa,oktalno!

.
.
.

Rješenje (oktalno):

1100	7775	7775
1101	CLEAR	7200
	STORE 1000	3200
	STORE 10	3010
	COMPL	7040
	ADD 1102	1302
	STORE 1102	3302
	CLEAR	7200
	COMPL	7040
	ADD 1103	1303
	STORE 1103	3303
	ISZ I 1116	2716
	JUMP 1101	5301
	HALT	7402
	1100	1100

.
.
.

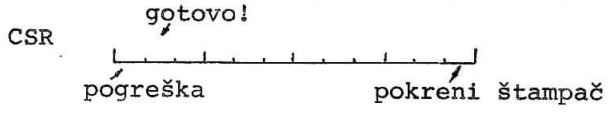
Mem. lokacije koje će biti promijenjene (bit će upisana 0):

(1000) = 0 (10) = 0
(177) = 0 (7) = 0
(176) = 0 (6) = 0

106. Ispitni uredjaj za praćenje rada računala prekida u pravilnim vremenskim razmacima tok programa i objavljuje tog trenutka zatečenu adresu izvršene naredbe, "kod" same naredbe i sadržaj akumulatora. To ispisuje na 3 numerička displeja (adresu pomoću naredbe WR DISADR , "kod" naredbe pomoću WR DISMEM i sadržaj akumulatora pomoću WR DISAK. Svaki displej objavljuje podatak u 12 bitnom kodu. Upotpuni zadani potprogram da se postigne željena funkcija !

```
0           0
            STORE AK
            CLEAR
            COMPL
            ADD 0
            STORE ADRESA
            WR DISADR
            CLEAR
            ADD I ADRESA
            AND MASKA
            WR DISMEM
            CLEAR
            ADD AK
            WR DISAK
            ION
            JUMP I 0
AK          0
MASKA       7000
ADRESA      0
```

107. U štampač (adresa 10) treba prenijeti programiranim uvjetnim prijenosom 100_8 znakova iz memorijskog polja s početnom adresom 2000_8 . Radom štampača upravlja se preko komandnog registra CSR. Ako se pojavila pogreška treba skočiti na lokaciju GRESKA! Računalo ima organizaciju kao PDP-8 , a instrukcioni set školskog računala.
Potrebno je napisati program koji omogućava prijenos podataka iz računala u štampač !



Rješenje:

```
        *5
P       CLEAR
        ADD A2000
        STORE PO
        CLEAR
        ADD M100
        STORE COUNT
NEXT    CLEAR
        IAC
        WRCSR
        CLEAR
        ADD I PO
        WR10
PON     RDCSR
        ROTL
        SKIPL
        SKIP
        JUMP GRESKA
        SKIPN
        JUMP PON
        ISZ PO
        ISZ COUNT
        JUMP NEXT
        HALT
A2000   2000
M100    7700
PO       0
COUNT    0
          .
GRESKA    .
          .
```

108. Računalo komunicira s jedinicom magnetske trake preko posebnog komandnog registra čija je adresa RKOM. Trenutno stanje jedinice mag. trake dobiva računalo preko "status registra" (adresa RSTAT).

Registri RKOM i RSTAT su 8-bitni ,a računalo je 12 bitno.

ČITAJ	PIŠI	PREMOTAJ NA POČETAK	ZAUSTAVI	1 BLOK NAPRIJED	X	X	IZVRŠI KOMANDU

RKOM

GREŠKA	X	X	X	JED. MAG. TRAKA JE ZAUZETA	X	X	KOMANDA JE IZVRŠENA

RSTAT

Treba napisati program koji će najprije premotati traku na početak, a zatim pomaknuti traku za 10_{10} blokova naprijed. Ako je jedinica mag. traka zauzeta, treba čekati da se oslobodi, a ako se pojavila greška, treba isti trenutak zaustaviti program !

Rješenje:

```
*201
P1      RDRSTAT
        AND K1
        SKIPZ
        JUMP .-3
        ADD K2
        WRRKOM
        RDRSTAT
        ROTL
        SKIPL
        SKIP
        JUMP KRAJ
        AND K3
        SKIPZ
        SKIP
        JUMP .-10
        CLEAR
        ADD DESET
        COMPL
        IAC
        STORE M10
                                  *
DX      CLEAR
        ADD K4
        WRRKOM
        RDRSTAT
        ROTL
        SKIPL
        SKIP
        JUMP KRAJ
        AND K3
        SKIPZ
        SKIP
        JUMP .-10
        ISZ M10
        JUMP DX
KRAJ    HALT
K1      0010
K2      0041
K3      0002
K4      0011
M10     7766
DESET   0012
```

4.2 KOMERCIJALNO MINIRAČUNALO PDP-8

(G. Smiljanić : Osnove digitalnih računala,

(poglavlje 8.7))

109. Napraviti kratak test program za računalo PDP-8e. Program treba uvećavati sadržaj akumulatora za 1. (Rezultat rada može se vidjeti na konzoli računala, ako se prikazuje trenutni sadržaj akumulatora).

Rješenje:

adresa	okt. sadržaj	PDP-8 simb.	ŠK. RAČ. simb.	
0000	7001	IAC	IAC	
0001	2101	ISZ 101	ISZ 101	} petlja kašnjenja da bi sadržaj akumulatora bio vidljiv
0002	5001	JMP 1	JUMP 1	
0003	5000	JMP 0	JUMP 0	
.	.	.	.	
.	.	.	.	
0101	0000	0	0	

110. Analizirati kratki program za testiranje prijemnika teleprintera (TTY) pomoću računala PDP-8e !

adresa	sadržaj (okt.)	PDP-8 simb.	Rješenje: (ŠK. RAČ.) simb.
0000	6032	KCC	CLTTY CLEAR
0001	6031	KSF	SKPTTY
0002	5001	JMP 1	JUMP 1
0003	6036	KRB	RDTTY CLTTY
0004	5001	JMP 1	JUMP 1

rezultat: Tastatura teleprintera će se nakon pritisnute tipke osloboditi i omogućiti novo utipkavanje !

111. Analizirati kratki program za testiranje predajnika teleprintera (TTY) pomoću računala PDP-8e !

Rješenje:

adresa	sadržaj (okt.)	PDP-8 simb.	(ŠK. RAČ.) simb.	
0000	7604	LAS	CLEAR ADD S /	S = SWITCH REG.
0001	6046	TLS	WRTTY	
0002	6041	TSF	SKPTTY	
0003	5002	JMP 2	JUMP 2	
0004	5000	JMP 0	JUMP 0	
.	.	.	.	
.	.	.	.	

rezultat: teleprinter će štampati sadržaj "SWITCH" REGISTRA

112. Napisati program koji će pročitati jedan znak s tastature teleprintera i štampati ga na teleprinteru (EHO test). Program je smješten u memoriji računala PDP-8 od lokacije 0. Koristiti se rješenjima iz prethodnih primjera !

Rješenje:

adresa	sadržaj (okt.)	PDP-8 simb.	ŠK. RAČ. simb.	
0000	6032	KCC	CLTTY CLEAR	
0001	6031	KSF	SKPTTY	
0002	5001	JMP 1	JUMP 1	/ulaz
0003	6036	KRB	RDTTY CLTTY	
0004	6046	TLS	WRTTY	
0005	6041	TSF	SKPTTY	/izlaz
0006	5005	JMP 5	JUMP 5	
0007	5001	JMP 1	JUMP 1	

Rezultat: program će prihvatiti znak s tastature i štampati ga na teleprinteru (EHO TEST).

113. Napisati program za računalo PDP-8 koji će na teleprinteru štampati sve znakove po redu. Koristiti rješenja iz prethodnih primjera !

Rješenje:

adresa	okt. sadržaj	PDP-8	ŠK. RAČ.
0000	7001	IAC	IAC
0001	6046	TLS	WRTTY
0002	6041	TSF	SKPTTY
0003	5002	JMP 2	JUMP .-1
0004	5000	JMP 0	JUMP 0

rezultat: štampat će se svi znakovi za koje postoji ASCII ekvivalent (kod) !

114. Napisati program za upisivanje 0 u odredjene memorijske lokacije računala PDP-8e !

Rješenje:

adresa	okt. sadržaj		PDP-8		ŠK. RAČ.
0100	1306	POC,	TAD NULA	POC	ADD NULA
0101	3707		DCA I ADR		STORE I ADR
					CLEAR
0102	2307		ISZ ADR		ISZ ADR
0103	2310		ISZ COUNT		ISZ COUNT
0104	5300		JMP POC		JUMP POC
0105	7402		HLT		HALT
0106	0000	NULA,	0000	NULA	0
0107	0200	ADR,	0200	ADR	0200
0110	7601	COUNT,	7601	COUNT	7601 /-177_8

rezultat: program će upisati 0 u 177_8 lokacija počevši od mem. adrese 200_8 !

115. Napisati program za upisivanje 0 u odredjene memorijske lokacije računala PDP-8e. Potrebno je iskoristiti svojstvo mem. lokacije 10_8 koja automatski uvećava svoj sadržaj za 1 kod indirektnog adresiranja preko te lokacije !

Rješenje:

adresa	okt. sadržaj	PDP-8	ŠK. RAČ.
.	.	.	.
0004	1007	TAD 7	ADD 7
0005	3410	DCA I 10	STORE I 10
			CLEAR
			ISZ 10
0006	5004	JMP 4	JUMP 4
0007	0000	0	0
0010	0011	11	11

rezultat: program će izbrisati cijelu memoriju počevši od lokacije 11_8 .

4.3 MIKRORAČUNALA

(U ovom dijelu dani su zadaci i rješenja za nekoliko različitih tipova 8 bitnih mikroračunala.
Priložene su izlazne liste koje generiraju asembleri tih mikroračunala, a u nekim primjerima i rezultati izvodjenja pojedinih programa.)

(FAIRCHILD F8)

116. Analizirati potprogram za čitanje jednog znaka s U/I jedinice 4 (port 4 s 8 ulazno/izlaznih linija). Preko porta 5 odredjuje se stanje vanjske jedinice. Napisati isti program za školsko računalo !

```
C
0000                    **********
0001                    * CITANJE
0002                    **********
0003                    *
0004   ADR.  MAŠ. KOD          ORG H'4600'
0005 4600 70                   LIS 0          Ø ——→ A
0006 4601 B5                   OUTS 5         (A) ——→ U/I PORT 5
0007 4602 A5            TK     INS  5         (U/I PORT 5) ——→ A
0008 4603 91 03                BM   R         AKO JE (A) < Ø SKOK NA R
0009 4605 90 FC                BR   TK        SKOK NA TK
000A 4607 A4            R      INS 4          (U/I PORT 4) ——→ A
000B 4608 57                   LR  7,A        (A) ——→ REG. 7
000C 4609 1C                   POP            POVRATAK IZ POTPROGRAMA
000D                           END
00
```

BROJ POGREŠAKA KOD ASEMBLIRANJA

ŠK. RAČ.

```
*4600
POC     Ø
        CLEAR
TK      RD 5
        SKIPN
        JUMP TK
R       RD 4
        STORE 7
        JUMP I POC
        END
```

117. Analizirati potprogram za pisanje jednog znaka iz registra 5 na U/I linije (adresa 4). Preko sadržaja porta 5 (8 U/I linija) odredjuje se stanje vanjske jedinice !

```
C
0000                    **********
0001                    * PISANJE
0002                    **********
0003                    *
0004                         ORG H'4700'
0005 4700 A5            TT   INS 5
0006 4701 13                 SL  1
0007 4702 91 03              BM  W
0008 4704 90 FB              BR  TT
0009 4706 47            W    LR  A,7
000A 4707 B4                 OUTS 4
000B 4708 1C                 POP
000C                         END
00

TT   4700  W    4706
PUNCH
```

118. Potprogram RC čita jedan znak s teleprintera, a potprogram WC piše jedan znak na teleprinter.

Potrebno je napisati program za učitavanje i ispisivanje jednog znaka na teleprinteru (EHO program)!

```
0000                    ***************
0001                    * PROGRAM EHO
0002                    ***************
0003                    *
0004                    RC   EQU H'4600'
0005                    WC   EQU H'4700'
0006                    *
0007                         ORG H'5000'
0008                    *
0009 5000 28 46 00 KRUG PI   RC
000A 5003 28 47 00      PI   WC
000B 5006 90 F9         BR   KRUG
000C                    END
00
```

119. Analizirati potprogram koji pretvara jedan ASCII znak (koji se nalazi u registru 7) u heksadecimalni broj (rezultat spremi u registar 7)!

```
0000                    **********
0001                    * ASCII
0002                    **********
0003                    *
0004                         ORG H'4500'
0005 4500 47             LR  A,7
0006 4501 25 09          CI  H'09'
0007 4503 82 03          BC  AS
0008 4505 24 07          AI  C'A'-C'9'-1
0009 4507 24 30     AS   AI  C'0'
000A 4509 57             LR  7,A
000B 450A 1C             POP
000C                     END
```

120. Analizirati program koji čita sve znakove s teleprintera i sprema ih u memoriju računala. U trenutku kad računalo prepozna znak "@" , štampaju se svi prethodno upisani znakovi s tastature teleprintera !

```
0000                    ******************************
0001                    * UPIS/ISPIS BLOKA PODATAKA
0002                    ******************************
0003                    *
0004                    RC    EQU H'4600'
0005                    WC    EQU H'4700'
0006                    *
0007                          ORG H'5100'
0008                    *
0009 5100 2A 55 00 PP   DCI   H'5500'
000A 5103 28 46 00 NN   PI    RC
000B 5106 47            LR    A,7
000C 5107 17            ST
000D 5108 25 40         CI    C'@'
000E 510A 94 F8         BNZ   NN
000F 510C 2A 55 00      DCI   H'5500'
0010 510F 16        SS  LM
0011 5110 25 40         CI    C'@'
0012 5112 84 ED         BZ    PP
0013 5114 57            LR    7,A
0014 5115 28 47 00      PI    WC
0015 5118 90 F6         BR    SS
0016                    END
00

NN   5103   PP   5100   RC   4600   SS   510F   WC   4700

PUNCH
```

121. Analizirati program za zbrajanje N prirodnih brojeva!
Koliko lokacija memorije zauzima zadani program ?

```
0000                    ****************************
0001                    * ZBROJ BROJEVA OD 0 DO N
0002                    ****************************
0003                    *
0004                    WC    EQU H'4700'
0005                    WB    EQU H'4800'
0006                    AS    EQU H'4500'
0007                    *
0008                          ORG H'5200'
0009                    *
000A 5200 2A 56 00            DCI H'5600'
000B 5203 16                  LM
000C 5204 51                  LR  1,A
000D 5205 57                  LR  7,A
000E 5206 28 45 00            PI  AS
000F 5209 28 47 00            PI  WC
0010 520C 28 48 00            PI  WB
0011 520F 28 48 00            PI  WB
0012 5212 41                  LR  A,1
0013 5213 31              DD  DS  1
0014 5214 84 04               BZ  ISP
0015 5216 C1                  AS  1
0016 5217 90 FE               BR  DD
0017 5219 52              ISP LR  2,A
0018 521A 14                  SR  4
0019 521B 57                  LR  7,A
001A 521C 28 45 00            PI  AS
001B 521F 28 47 00            PI  WC
001C 5222 42                  LR  A,2
001D 5223 21 0F               NI  H'0F'
001E 5225 57                  LR  7,A
001F 5226 28 45 00            PI  AS
0020 5229 28 47 00            PI  WC
0021 522C 29 00 1D            JMP H'001D'
0022                          END
00
0000                    ************************
0001                    * ISPIS PRAZNOG MJESTA
0002                    ************************
0003                    *
0004                          ORG H'4800'
0005                    *
0006 4800 20 20               LI  C' '
0007 4802 57                  LR  7,A
0008 4803 A5              TT  INS 5
0009 4804 13                  SL  1
000A 4805 91 03               BM  W
000B 4807 90 FB               BR  TT
000C 4809 47              W   LR  A,7
000D 480A B4                  OUTS 4
000E 480B 1C                  POP
000F                          END
```

(INTEL I8080)

122. Potprogram CNVBN pretvara sadržaj C registra (ASCII karakter "0" -"9" ili "A" - "F" u heksadecimalan broj $0 - F_{16}$ koji se nalazi u akumulatoru. Analiziraj zadani potprogram !

```
CNVBN:
        MOV A,C          / (C)    A
        SUI "0"          / (A) ← "0"
        CPI 10           / ako je rezultat 0-9 10
        RM               / povratak iz potprograma
        SUI 7            / rezultat je izmedju 17 10 i 23 10
        RET              / zato treba oduzeti 7
```

123. Potprogram čita 2 ASCII znaka s tastature teleprintera i pretvara ih u 2 heksadecimalna broja (1 bajt). Konačan rezultat nalazi se u akumulatoru. Analizirati potprogram za mikroprocesor I8080 koji to radi. Koriste se gotovi potprogrami CNVBN koji pretvara ASCII znak u heksadecimalni i RICH koji čita jedan ASCII karakter !

```
BYTE:   PUSH B          / spremi (B) u "STACK"
        CALL RICH       / pročitaj ASCII znak
        MOV C,A
        CALL CNVBN      / pretvori ga u heksadec. vrijednost
        RLC
        RLC
        RLC
        RLC             /pomakni ga u gornja 4 bita
        MOV B,A
        CALL RICH       / pročitaj drugi ASCII znak
        MOV C,A
        CALL CNVBN      / heksadec. vrijednost
        ORA B           / donja 4 bita
        MOV A,B         / rezultat u akumulator
        POP B           / vrati (B) iz "STACK"-a
        RET             / povratak iz potprograma
```

(INTEL I8085)

124. Opiši zadani program za mikroračunalo I8085. Koja je njegova funkcija ?

```
START:  LXI H,BTBL      /registri H i L će
GTBIT:  RAR             /pokazivati na tabelu BTBL
        JC GETAD
        INX H           / (H,L) = (H,L) + 2
        INX H
        JMP GTBIT
GETAD:  MOV E,M         /postavljeni bit u akumulato-
        INX H           /ru je pronadjen. Stavi adresu
        MOV D,M         /skoka u D,E registarski par
        XCHG            /zamijeni reg. parove DE i HL
        PCHL            / skoči na zadanu adresu
         .
         .
         .
BTBL    DW ROUT1        / tabela adresa na koje treba
        DW ROUT2        / skočiti ovisno o postav-
        DW ROUT3        / ljenom bitu u akumulatoru.
        DW ROUT4        / Svaka adresa sastoji se
        DW ROUT5        / iz dva bajta !
        DW ROUT6
        DW ROUT7
        DW ROUT8
```

Rezultat: programski odsječak START: koristi H i L reg. kao pokazivač na tabelu adresa BTBL, ovisno o sadržaju akumulatora. Programski dio GETAD prenosi adresu iz tabele BTBL u H,L reg. par preko D,E registarskog para i pomoću instrukcije PCHL skače na odgovarajuću adresu (ROUT1 ...ROUT8).

125. Prenesi 5 podataka iz polja s početnom adresom 260_{16} u polje s početnom adresom 900_{16} . Napiši program za mikroprocesor I8085 !

Rješenje:

```
        .
        .
        .
        MVI B,5H        /postavi brojač
        LXI H,260H      /pripremi početnu adresu u HL reg.
        LXI D,900H      / stavi 900H u DE reg. par
PETLJA: MOV A,M         / stavi podatak u akumulator
        STAX D          / spremi ga na adresu (DE)
        INX D           / (DE) = (DE)+1
        INX M           / (HL) = (HL)+1
        DCR B           / (B) = (B)-1
        JNZ PETLJA      / ako (B) nije 0 skok na PETLJA
        HLT             / ako (B)=0 zaustavi program
        .
        .
        .
```

126. Intel 8279 je upravljački sklop za heksadecimalni prikaz podataka na "displeju" i učitavanje podataka preko tastature. Na mikroprocesor I8085 priključuje se direktno preko adresnih linija i linija podataka. Kontrolne riječi za rad s prikazom podataka šalju se preko adrese 1900H (H-heksadecimalno). Mogu biti slijedeće:

90H - ispisivanje 4 znaka koja slijede u adresni dio prikaza (4 lijeva polja)

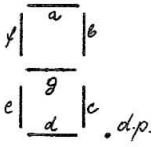
94H - ispisivanje 2 znaka koji slijede u dio prikaza za podatke (2 desna polja)

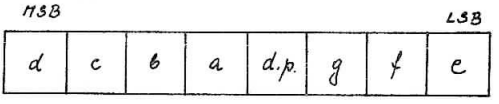
A3H - ispiši "prazno mjesto" u sva polja prikaza i zabrani pisanje

A0H - omogući daljnje ispisivanje

itd.

Znak koji treba biti ispisan, šalje se u RAM jedinice za ispis 8279 i to u obliku naloga za aktiviranje pojedinih segmenata znaka smještenog u jednu 8 bitnu riječ, a preko adrese 1800H :





Osam bitova riječi odredjuje stanje svakog dijela znaka i decimalne točke (log. "1" - segment je zatamnjen , log. "0" - segment je osvijetljen) , npr. za slovo "H" kod bi izgledao: $10011000_2 = 98H$.

Zadatak:

Analizirati program kojim će se ispisati poruka "HELP" u adresnom dijelu prikaza koji je preko jedinice 8279 vezan na osnovnu konfiguraciju mikroračunala MCS85 (INTEL 8085). Program počinje u lokaciji 2000H !

adresa.	sadržaj (heksa.)	simb.	komentar
2000	3E 90	MVI A,90H	/uzmi upravljačku
	32 00 19	STA 1900H	/riječ za ispis na
	3E 98	MVI A,98H	/adresnom dijelu
	32 00 18	STA 1800H	/ kod za "H" u RAM
	3E 68	MVI A,68H	/kod za "E" u A
	32 00 18	STA 1800H	/ kod za "E" u RAM
	3E 7C	MVI A,7CH	/
	32 00 18	STA 1800H	/ kod za "L" u RAM
	3E C8	MVI A,C8H	/
	32 00 18	STA 1800H	/ kod za "P" u RAM
	3E 94	MVI A,94H	/ upravljačka riječ
	32 00 19	STA 1900H	/ za ispis podatka
	3E FF	MVI A,FFH	/ kod za " " u A
	32 00 18	STA 1800H	/ kod za " " u RAM
	3E FF	MVI A,FFH	/ za prikaz podatka
	32 00 18	STA 1800H	/
2028	76	HLT	/ stop

127. Analizirati program koji će paliti i gasiti zapis "HELP" iz prethodnog zadatka !

Rješenje: početak programa jednak je rješenju iz prethodnog zadatka. Nastavak programa je:

```
                                  .
2028₁₆   00          NOP          / umjesto HLT iz prethod-
2029     11 FF FF    LXI D,FFFFH    /nog primjera
202C     1B          DCX D
         7A          MOV A,D      / vremenska petlja
         B3          ORA E        / od FFFFH puta
         C2 2C 20    JNZ 202CH
         3E A3       MVI A,A3H    / upravljačka riječ za
         32 00 19    STA 1900H    / upis " "
         11 FF FF    LXI D,FFFFH
203A     1B          DCX D        / vremenska petlja od
         7A          MOV A,D      / FFFFH puta
         B3          ORA E
         C2 3A 20    JNZ 203AH
         3E A0       MVI A,A0H    / upravljačka riječ za
         32 00 19    STA 1900H    / dozvolu novog zapisa
2047     C3 00 20    JMP 2000H    / vrati se na početak
                                  .
                                  .
                                  .
```

(ZILOG Z80)

128. Analizirati što radi zadani program koji se izvodi na jednom 8-bitnom računalu s dva registra A i B.
Napisati program za školsko računalo koji obavlja isti zadatak !

	program:	komentar:
ORG	0H	početak od lokacije 0H
	LD A,(100H)	(100H) → A
	LD B,A	(A) → B
	XOR A	(A)⊕ (A) → A (⊕ je EX ILI)
AA	ADD A,B	(A) + (B) → A
	DEC B	(B) -1 → B
	JR NZ,AA-$	ako je (B) ≠ 0 skoči na lokaciju AA, inače nastavi program
	LD (101H),A	(A) → 101H
	HALT	

Rješenje: Program zbraja prvih N cijelih brojeva !

Program za školsko računalo :

```
        .
        .
PK   CLEAR
     ADD 400
     COMPL
     IAC
     STORE COUNT
     CLEAR
     ADD COUNT
     ISZ COUNT
     JUMP .-2
     COMPL
     IAC
     STORE 401
     HALT
```

129. Usporediti zadani program koji se izvodi na mikroračunalu Z80 s programom iz prethodnog zadatka. Analizirati pojedine instrukcije i broj bajtova koji one zauzimaju !

```
.E :AS
OPTIONS?
READY PASS2?


   ADDR  OBJECT    ST #

                   0001 ;***********************************
                   0002 ; SUMA N CIJELIH BROJEVA            *
                   0003 ;***********************************
                   0004 ; ID. VJEZBE ER II 18.12.1978.
                   0005 ;
                   0006   ORG 0H
  '0000  3A0001    0007         LD A,(100H)
  '0003  47        0008         LD B,A
  '0004  AF        0009         XOR A
  '0005  80        0010 AA      ADD A,B
  '0006  05        0011         DEC B
  '0007  20FC      0012         JR NZ,AA-$
  '0009  320101    0013         LD (101H),A
  '000C  76        0014         HALT
                   0015         END

ERRORS=0000
ERRORS=0000
.L 0
BEG ADDR 0000
EXECUTE  0000
END ADDR 000C
UNDEF SYM 00
*.
.M 100
0100 03
0101 00 .
.
.E 0

.M 100
0100 03
0101 06 .  REZULTAT
.
.
```

130. Analizirati zadani program za vremensko kašnjenje.
Koliko se dugo izvršava zadani program ?

```
 ADDR   OBJECT     ST #

                   0001 ;************************
                   0002 ; VREMENSKO KASNJENJE    *
                   0003 ;************************
                   0004 ; I.D. M.ZAGAR  131278.
                   0005 ;
                   0006   ORG 0H
'0000   018813     0007           LD BC,5000D  *
'0003   C5         0008 KAS       PUSH BC
'0004   06BB       0009           LD B,187
'0006   10FE       0010 KAS1      DJNZ KAS1-$
'0008   C1         0011           POP BC
'0009   0B         0012           DEC BC
'000A   78         0013           LD A,B
'000B   B1         0014           OR C
'000C   20F5       0015           JR NZ,KAS-$
'000E   76         0016           HALT
                   0017 ;
                   0018           END

ERRORS=0000
ERRORS=0000
.
.
.
```

RJEŠENJE :

* PROGRAM SE IZVRŠAVA 5000 msek. = 5 SEK.

131. Program prenosi 100_{10} podataka iz bloka ADR1 u blok ADR2. Koristi se instrukcija "LDIR" koja ima slijedeće svojstvo nakon svakog izvodjenja :

(HL) → (DE), (HL)+1 → HL , (DE)+1 → DE, (BC)-1→BC, ako je (BC)≠ 0 onda (PC)-2 → PC !

Rješenje:

```
•E :AS
OPTIONS?
READY PASS2?
```

```
   ADDR  OBJECT    ST #

                   0001 ;*********
                   0002 ;PRIJENOS BLOKA MEMORIJE
                   0003 ;*********
                   0004 ;ID. M.ZAGAR 131278.
                   0005 ;
   >0000           0006 ADR1     EQU     0H
   >1000           0007 ADR2     EQU     1000H
                   0008          ORG 100H
                   0009 ;
  '0100  210000    0010          LD HL,ADR1
  '0103  110010    0011          LD DE,ADR2
  '0106  016400    0012          LD BC,100D
  '0109  EDB0      0013          LDIR
  '010B  76        0014          HALT
                   0015 ;
                   0016          END

ERRORS=0000
ERRORS=0000
•
```

132. Potrebno je prenijeti blok podataka s početnom adresom 200_{16} u blok s početnom adresom 300_{16}. Kraj bloka odredjuje znak "$". Maksimalna dužina bloka je 100_{10} podataka. Analiziraj program koji to radi !

```
 ADDR   OBJECT     ST #

                   0001 ;****************************
                   0002 ;  PRIJENOS PODATAKA         *
                   0003 ;****************************
                   0004 ;I.D.VJEZBE RADIO KOMUNIKACIJE, 20.12.1978.
                   0005 ;
                   0006  ORG 100H
 '0100  210002     0007          LD HL,200H
 '0103  110003     0008          LD DE,300H
 '0106  016400     0009          LD BC,100D
 '0109  3E24       0010          LD A,'$'
 '010B  BE         0011 PETLJA:  CP (HL)
 '010C  2805       0012          JR Z,KRAJ-$
 '010E  EDA0       0013          LDI
 '0110  EA0B01'    0014          JP PE,PETLJA
 '0113  76         0015 KRAJ:    HALT
                   0016          END

EPRORS=0000
ERRORS=0000
```

133. Analizirati što radi zadani program ?

```
                        0001 ;***************************************
                        0002 ; POMICANJE ZA 16 BCD BROJEVA          *
                        0003 ;***************************************
                        0004 ; ID.  M.ZAGAR  261278
                        0005 ;
                        0006        ORG 50H
  '0050  2A5C00'        0007        LD HL,(POD)
  '0053  0610           0008        LD B,16D
  '0055  AF             0009        XOR A
  '0056  ED6F           0010 ROTAT: RLD
  '0058  23             0011        INC HL
  '0059  10FB           0012        DJNZ ROTAT-$
  '005B  76             0013        HALT
  '005C  0001           0014 POD    DEFW  100H
                        0015        END

ERRORS=0000
ERRORS=0000
.
.L 0
BEG ADDR 0050
EXECUTE  0050
END ADDR 005D
UNDEF SYM 00
*.
.M 100,11F
0100 01 02 03 04 05 06 07 08 09 01 02 03 04 05 06 07
0110 F8 76 41 00 41 00 41 00 41 00 41 00 41 00 41 00
..
.
.E 50    - izvrši program

.M 100,11F
0100 10 20 30 40 50 60 70 80 90 10 20 30 40 50 60 70
0110 F8 76 41 00 41 00 41 00 41 00 41 00 41 00 41 00
.E 50

.M 100,11F
0100 00 01 02 03 04 05 06 07 08 09 01 02 03 04 05 06
0110 F8 76 41 00 41 00 41 00 41 00 41 00 41 00 41 00
.LM
.
.
.M 100,11F
0100 12 34 56 78 90 01 12 23 34 45 56 67 78 89 90 00
0110 00 76 41 00 41 00 41 00 41 00 41 00 41 00 41 00
..
..
.E 50

.
.M 100,11F
0100 20 41 63 85 07 19 20 31 42 53 64 75 86 97 08 09
0110 00 76 41 00 41 00 41 00 41 00 41 00 41 00 41 00
```

Rješenje:

Program pomiče 16 BCD brojeva za jedno mjesto u lijevo !

a

b

134. Analizirati program za odbijanje dva BCD broja dužine 4 mjesta !

```
                        0001 ;****************************
                        0002 ; ODBIJANJE                  *
                        0003 ;****************************
                        0004 ; VJ. ER II  (AUTOMATIKA) 27.12.78.
                        0005 ;
                        0006          ORG 100H
'0100  210003           0007          LD HL,300H
'0103  111003           0008          LD DE,310H
'0106  0604             0009          LD B,4
'0108  A7               0010          AND A
'0109  1A               0011 SUBDEC:  LD A,(DE)
'010A  9E               0012          SBC A,(HL)
'010B  27               0013          DAA
'010C  77               0014          LD (HL),A
'010D  23               0015          INC HL
'010E  13               0016          INC DE
'010F  10F8             0017          DJNZ SUBDEC-$
'0111  76               0018          HALT
                        0019          END

ERRORS=0000
ERRORS=0000
```

```
•M 300
0300 26 1
0301 24 2
0302 37 3
0303 35 4
0304 45 0
0305 43 0
0306 13 0
0307 01 •
•M 310
0310 7F 2
0311 7F 3
0312 7F 4
0313 7F 6
0314 7F 0
0315 7F 0
0316 7F 0
0317 7F •
••
•M 300,320
0300 01 02 03 04 00 00 00 01 09 7F 7F 7F 7F 7F 7F 7F
0310 02 03 04 06 00 00 00 7F 7F 7F 7F 7F 7F 7F 7F 7F
0320 7F
•
•E 100

•
•M 300,31F
0300 01 01 01 02 00 00 00 01 09 7F 7F 7F 7F 7F 7F 7F
0310 02 03 04 06 00 00 00 7F 7F 7F 7F 7F 7F 7F 7F 7F
```

135. Slijedeći program množi dva 16-bitna broja koji se nalaze u registarskim parovima DE i HL. Rezultat se dobije u registarskom paru HL. Analizirati zadani program !

```
                        0001 ;*****************************
                        0002 ;MNOZENJE
                        0003 ;*****************************
                        0004 ;I D. VJ. ER 2  27. 12. 1978.
                        0005 ;
                        0006 ;;ULAZ : MULTIPLIKAND U  DE REG.
                        0007 ;         MULTIPLIKATOR U  HL REG.
                        0008 ;
                        0009 ;IZLAZ: REZULTAT U  HL REG.
                        0010 ;
                        0011          ORG 500H
'0500  0610             0012          LD B,16
'0502  4A               0013          LD C,D
'0503  7B               0014          LD A,E
'0504  EB               0015          EX DE,HL
'0505  210000           0016          LD HL,0
'0508  CB39             0017 MLOOP:   SRL C
'050A  CB1F             0018          RR A
'050C  3001             0019          JR NC,NOADD-$
'050E  19               0020          ADD HL,DE
'050F  EB               0021 NOADD:   EX DE,HL
'0510  29               0022          ADD HL,HL
'0511  EB               0023          EX DE,HL
'0512  10F4             0024          DJNZ MLOOP-$
'0514  76               0025          HALT
                        0026          END

ERRORS=0000
ERRORS=0000
.

.B 514
.E 500
0514 0044
.
.R 1
 PC   AF  I IF  BC   DE   HL  A'F' B'C' D'E' H'L'  IX   IY   SP
0514 0044 4000 0000 0002 0003 4000 4001 6161 F8E3 63E3 F263 FF90


.B 514
.E 500
0514 0045
.R 1
 PC   AF  I IF  BC   DE   HL  A'F' B'C' D'E' H'L'  IX   IY   SP
0514 0045 4000 0000 0000 0006 4000 4001 6161 F8E3 63E3 F263 FF90
.
```

136. Analizirati program za sortiranje podataka po veličini (broj podataka može biti izmedju 1 i 256_{10}) !

```
ADDR  OBJECT     ST #

                 0001 ;**********************************
                 0002 ;  SORTIRANJE
                 0003 ;**********************************
                 0004 ;ID.VJEZBE BR II    27.12.78.
                 0005 ;
                 0006 ;ULAZ : HL SADRZI ADRESU PODATAKA
                 0007 ;       C  SADRZI BROJ ELEMENATA KOJI CE BITI S
                 0008 ;       (1<C<256)
                 0009 ;
                 0010 ;IZLAZ : SORTIRANI PODACI
                 0011 ;
                 0012 ;REGISTRI KOJI SE KORISTE:
                 0013 ;  A,B,C,D,E,H,L,IX,IY
                 0014 ;
                 0015         ORG 0H
'0000  222600'   0016 SORT:   LD (DATA),HL
'0003  CB84      0017 LOOP:   RES FLAG,H
'0005  41        0018         LD B,C
'0006  05        0019         DEC B
'0007  DD2A2600' 0020         LD IX,(DATA)
'000B  DD7E00    0021 NEXT:   LD A,(IX+0)
'000E  57        0022         LD D,A
'000F  DD5E01    0023         LD E,(IX+1)
'0012  93        0024         SUB E
'0013  3008      0025         JR NC,NOEX-$
'0015  DD7300    0026         LD (IX),E
'0018  DD7201    0027         LD (IX+1),D
'001B  CBC4      0028         SET FLAG,H
'001D  DD23      0029 NOEX:   INC IX
'001F  10EA      0030         DJNZ NEXT-$
'0021  CB44      0031         BIT FLAG,H
'0023  20DE      0032         JR NZ,LOOP-$
'0025  76        0033         HALT
 >0000           0034 FLAG:   EQU 0
                 0035 DATA:   DEFS 2
                 0036         END

ERRORS=0000
ERRORS=0000
```

$(C) = 8_{10}$

```
.M 100,110
0100 01 02 03 04 06 05 05 08 07 00 FF 00 FF 00 FF 00
0110 FF
.
..
.E 0

.M 100,110
0100 08 06 05 05 04 03 02 01 07 00 FF 00 FF 00 FF 00
0110 FF
```